Airline Management System

DATE	**25-06-2025**
TEAM ID	**LTVIP2025TMID29751**
PROJECT NAME	**Airline Management System**
COLLEGE NAME	**Ideal Institute Of Technology**

1. **Team Leader:** Sabbavarapu Bhimesh Prasad
   Email : Bhimeshprasad15@gmail.com

2. **Team Member:** Yasaya Sai Kumar
   Email : saik89654@gmail.com

3. **Team Member:** Yedla Sai Srinivas
   Email :yedlasaisrinivas1@gmail.com

1. INTRODUCTION

1.1 Project Overview

The Airlines Management System is a Salesforce-based application designed to streamline the management of airline operations. It offers features for flight scheduling, passenger booking, crew management, and more, using a centralized platform to improve operational efficiency and customer satisfaction.

1.2 Purpose

The system aims to digitize and automate airline management, reducing manual errors, increasing transparency, and delivering better service through real-time updates and structured data handling.

2. IDEATION PHASE

2.1 Problem Statement

Managing an airline's day-to-day operations such as bookings, scheduling, and crew assignments manually is inefficient, error-prone, and lacks integration across teams.

2.2 Empathy Map

Canvas Focused on airline staff, passengers, and management to identify their needs and frustrations—highlighting the demand for real-time, error-free, and accessible systems.

2.3 Brainstorming

Salesforce was identified as the ideal platform for CRM integration, automation, and reporting capabilities. Objects

like Flights, Bookings, Passengers, and Crew were considered vital.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey

Map From searching flights to booking, checking-in, and boarding, the system ensures seamless transitions through each touchpoint.

3.2 Solution Requirement

Requirements include object management, custom fields, data validation, user roles, and visual dashboards.

3.3 Data Flow Diagram

Passenger -> Booking -> Flight/Crew assignments -> Check-in -> Reports

3.4 Technology Stack

Salesforce Platform

Apex (for logic)

Flow (for automation)

Reports & Dashboards

4. PROJECT DESIGN

4.1 Problem Solution

Fit Salesforce customization provides the flexibility needed to model airline operations effectively.

4.2 Proposed Solution

Custom Salesforce app named "Airlines Management System" with tailored objects, relationships, and user interfaces.

4.3 Solution Architecture

Objects interconnected via lookup relationships and custom tabs. Field-level validations enforced with Apex triggers.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Week 1–2: Ideation and setup

Week 3–4: Object and field creation

Week 5: Relationships and tabs

Week 6: Reports, dashboards

Week 7: Apex & flows

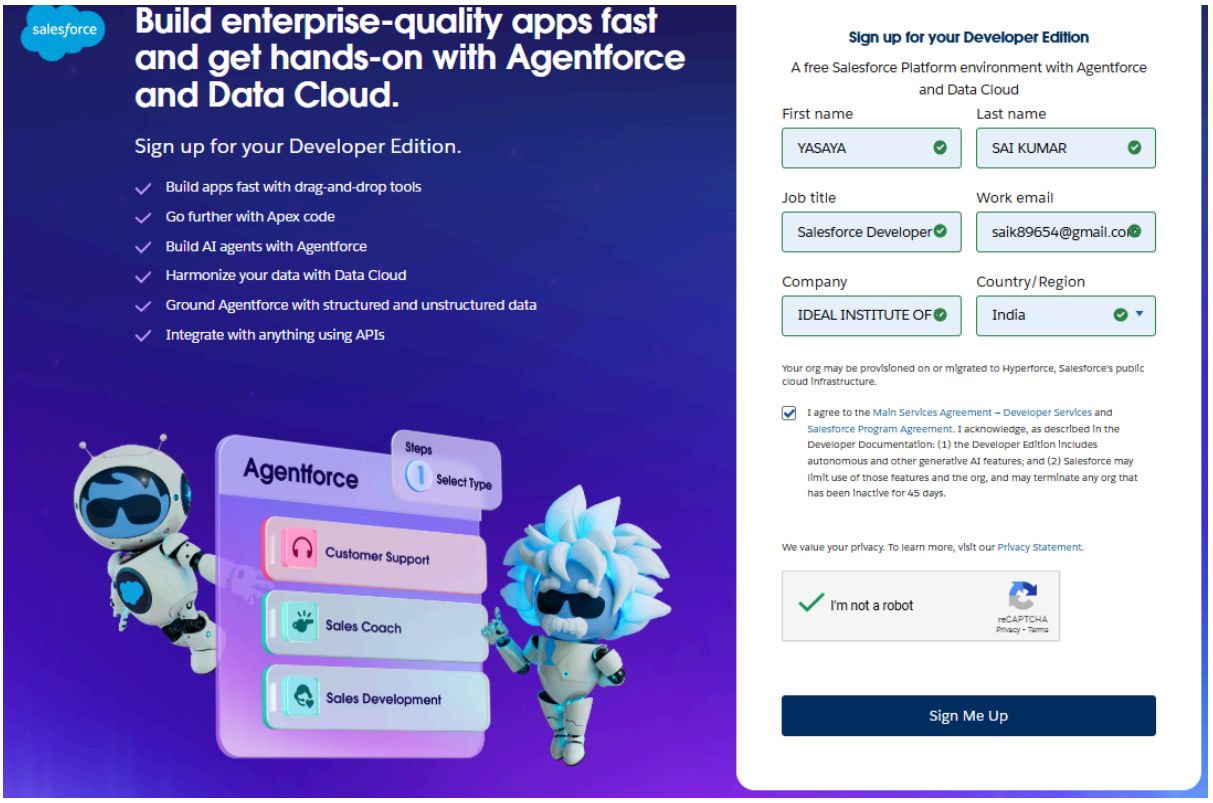
Week 8: Testing and final report

6. DEVELOPMENT PHASE (Salesforce Guided Project)

The development phase follows a structured, guided approach using Salesforce tools and best practices. It includes the following core milestones:

6.1 Developer Org Setup

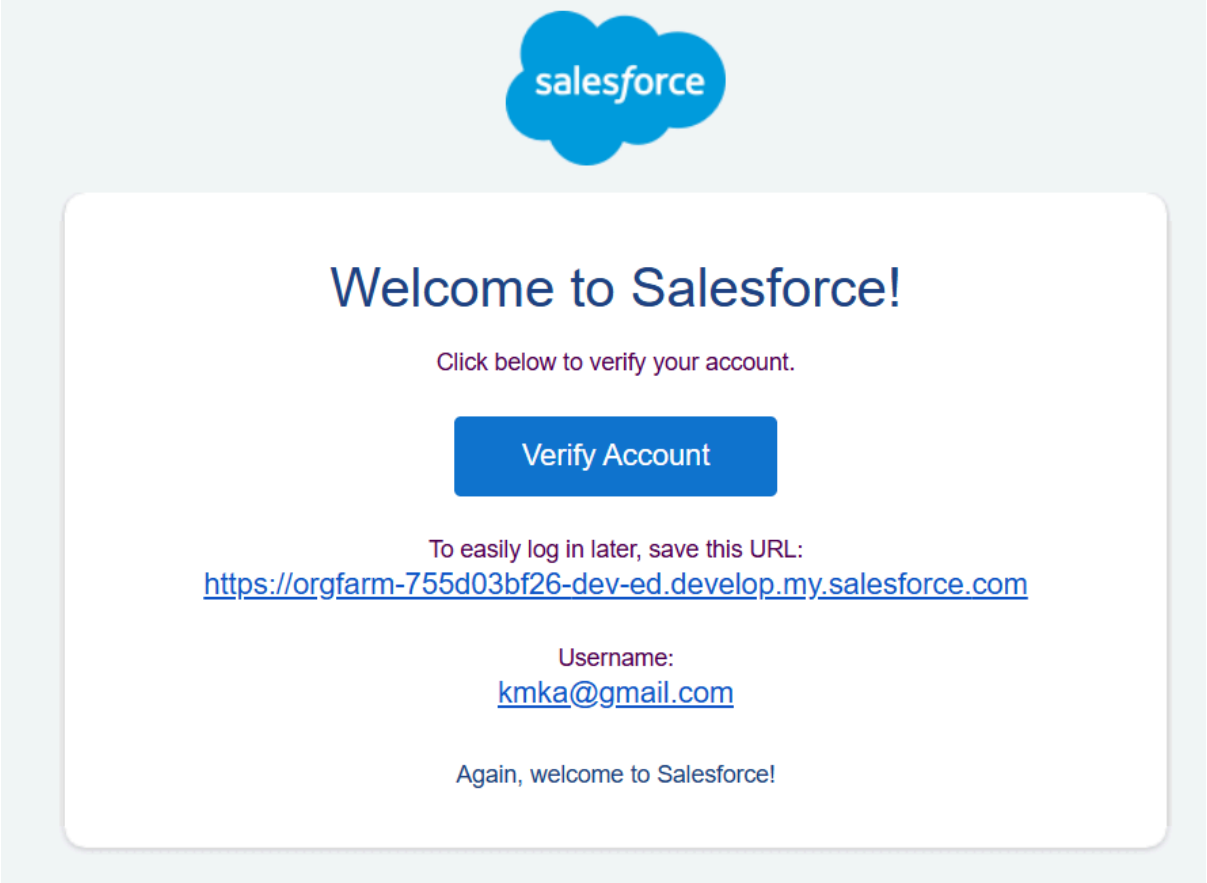
Created a Salesforce Developer Edition account via:developer.salesforce.com.



Configured basic org settings and verified user access.

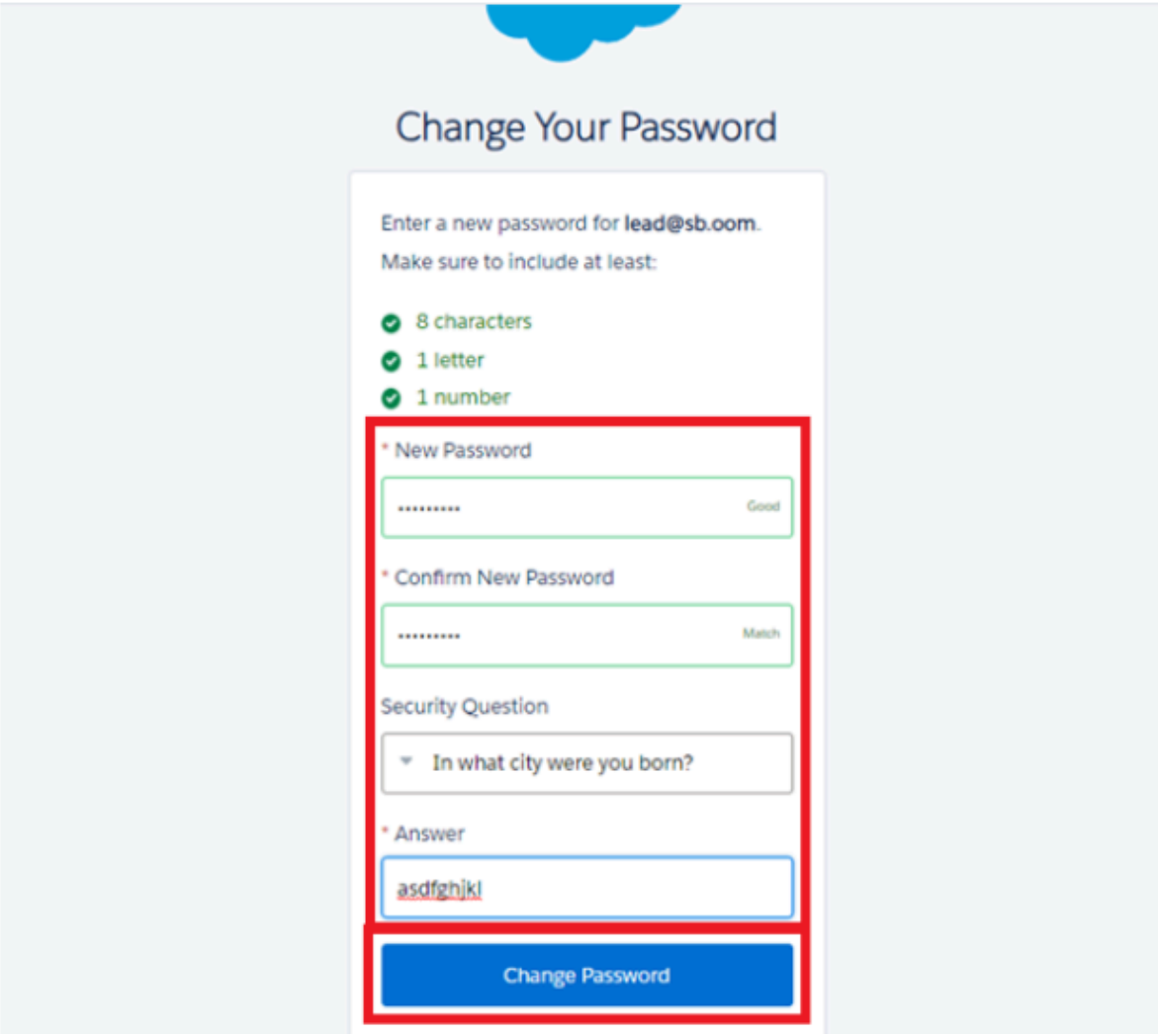
Account Activation

Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins.

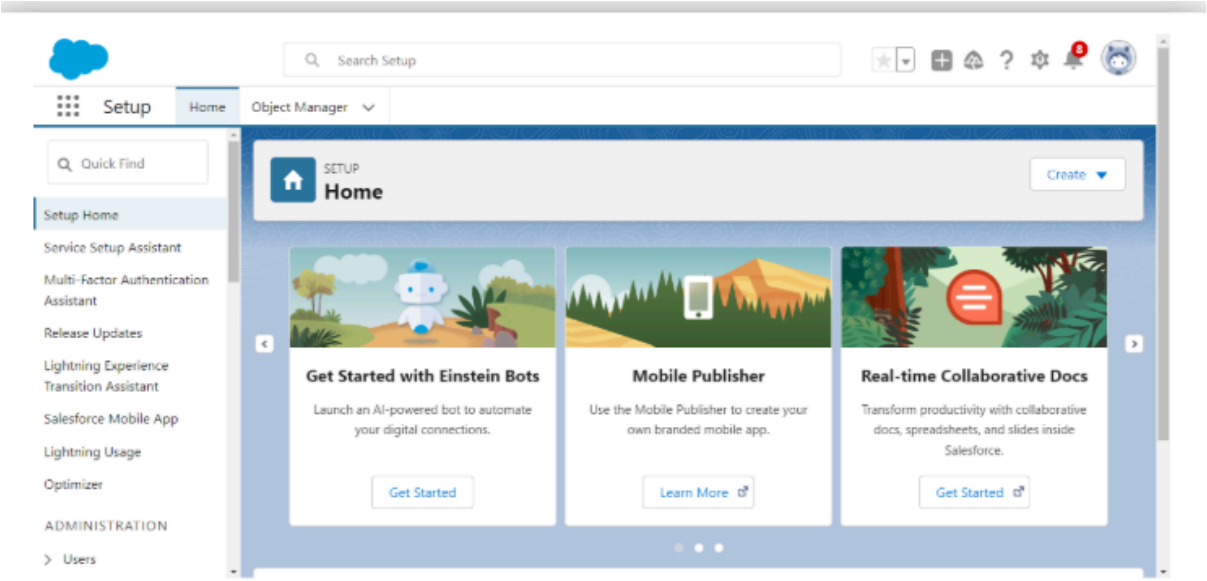


Clicked on Verify Account

Gave a password and answered a security question and clicked on change password.



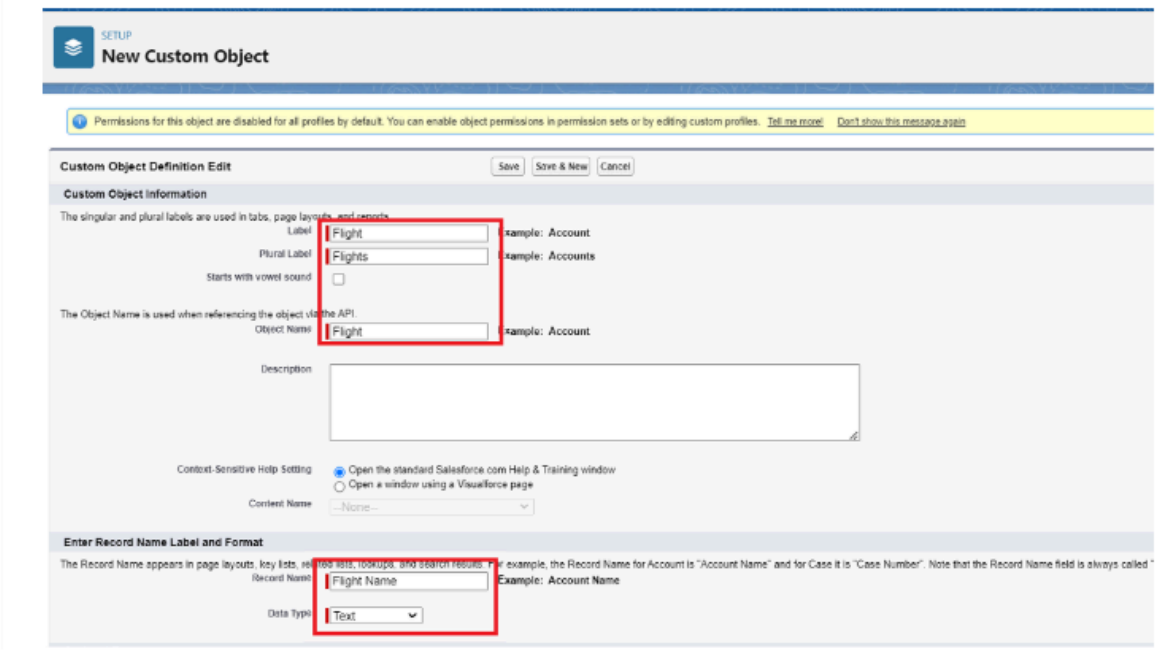
Then you will redirect to your salesforce setup page.

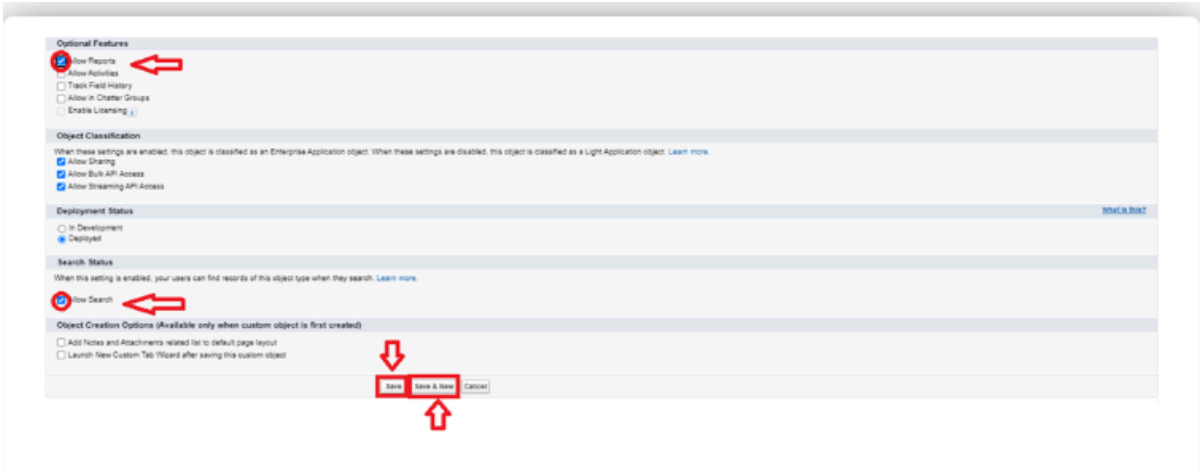


6.2 Custom Object Creation

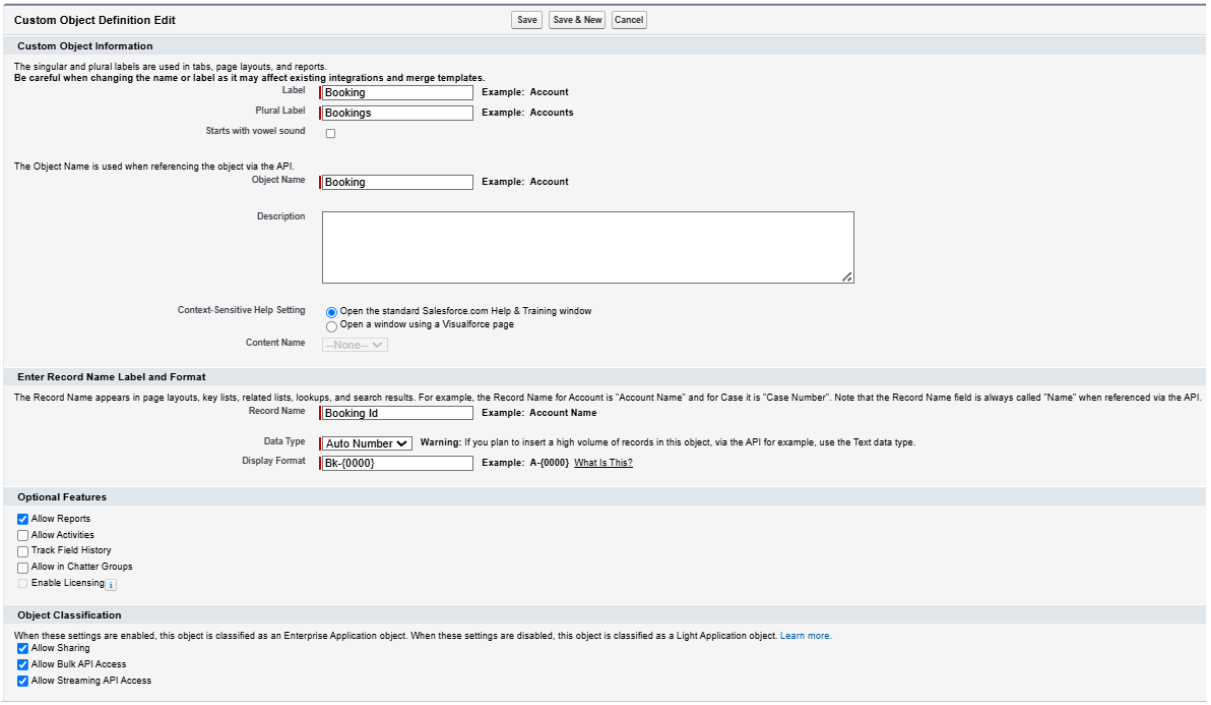
Created key custom objects:

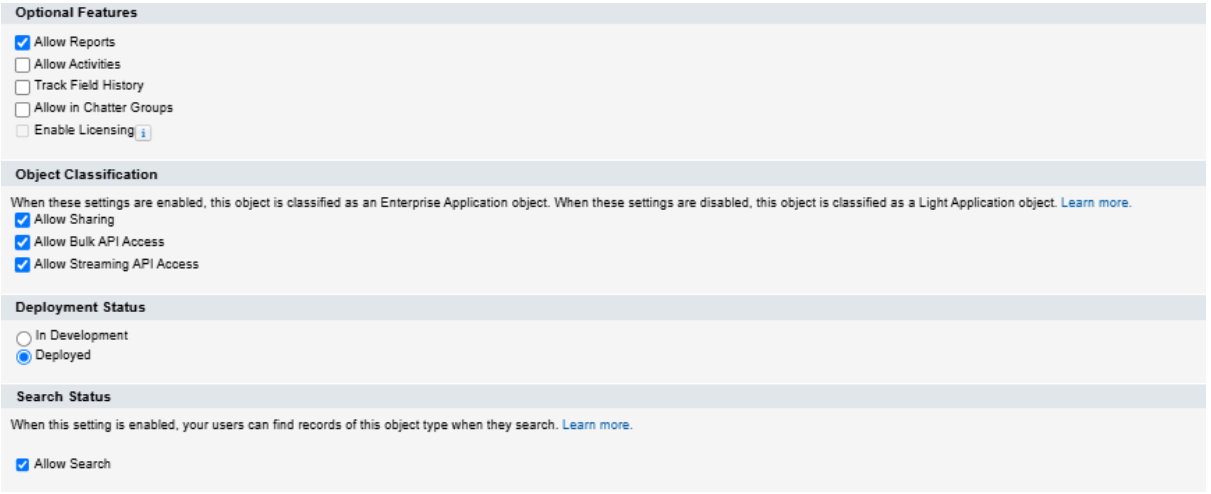
Flight – Stores flight details like name, capacity, departure time.





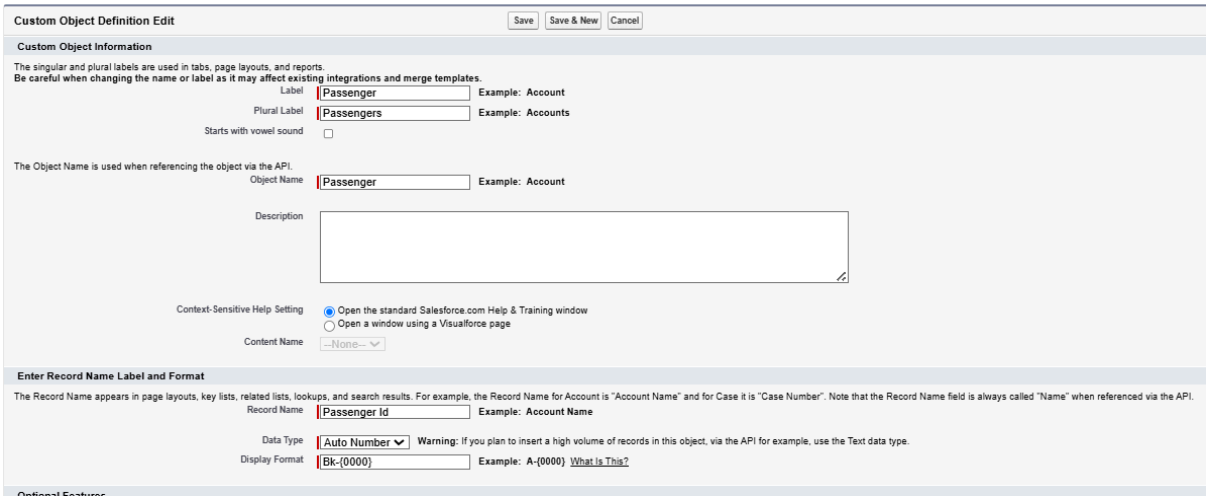
Booking – Handles reservations with fields like Booking ID, Departure Date.

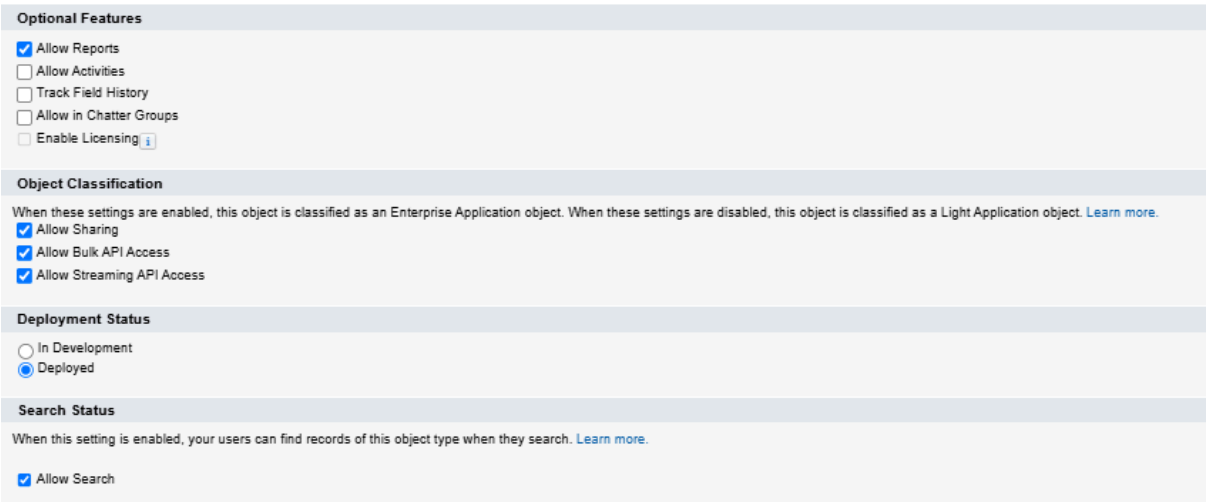




Passenger – Manages passenger details such as name, passport info.

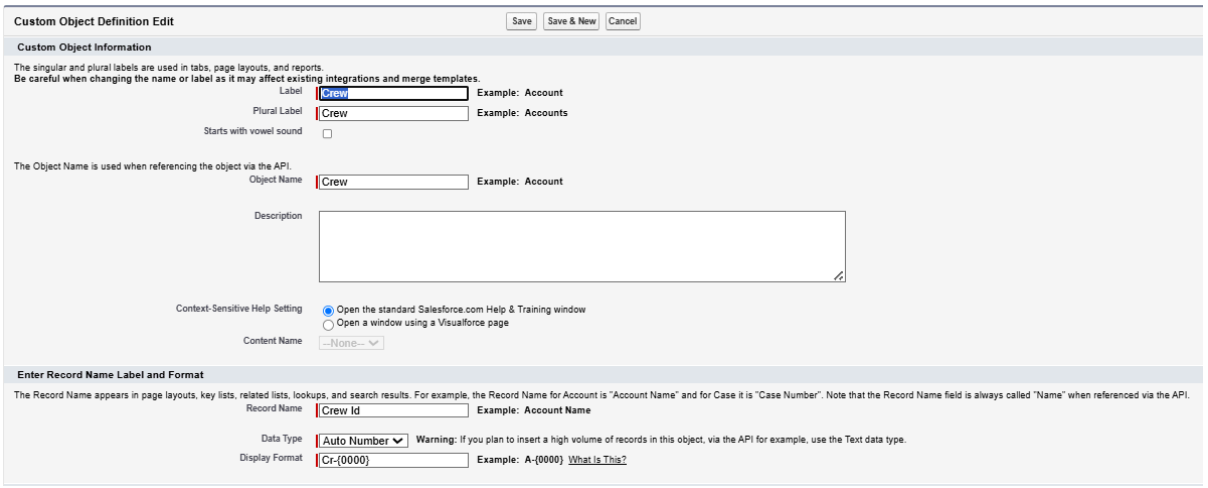
Passenger

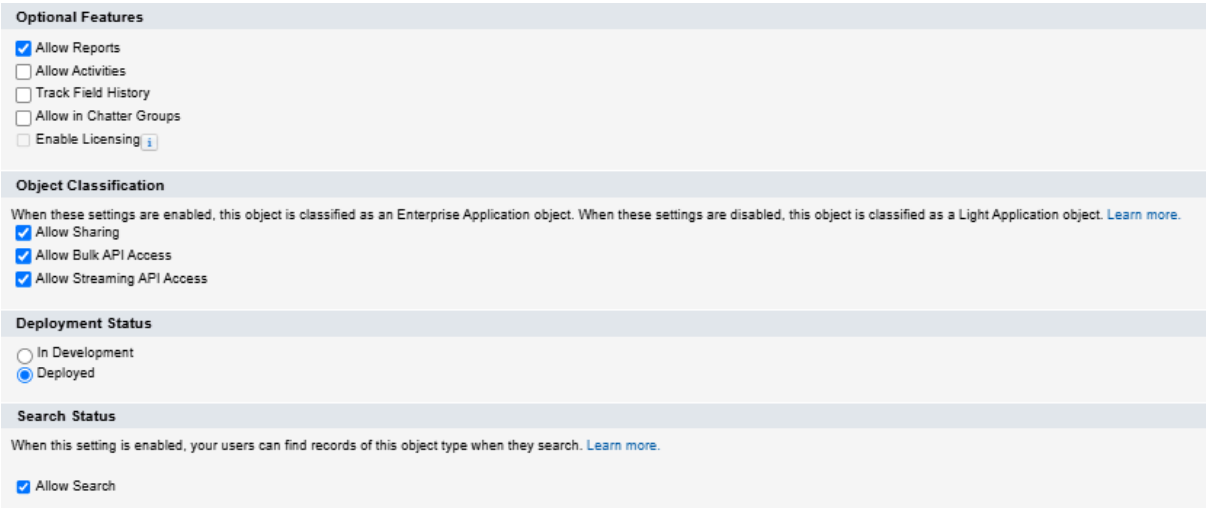




Crew – Records crew member roles and assignments.

Crew



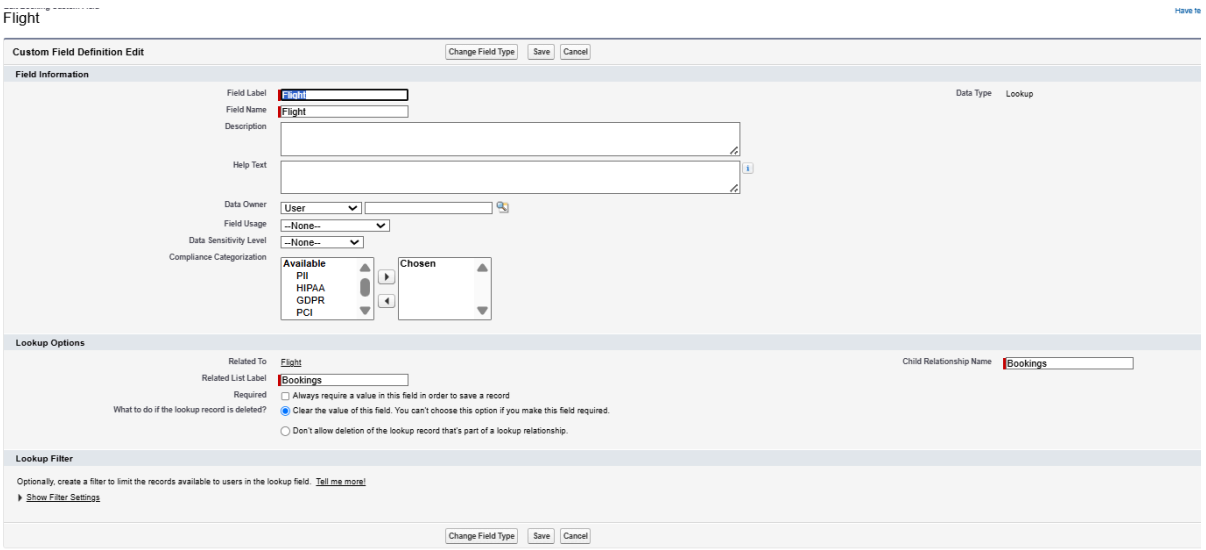


6.3 Field & Relationship Design

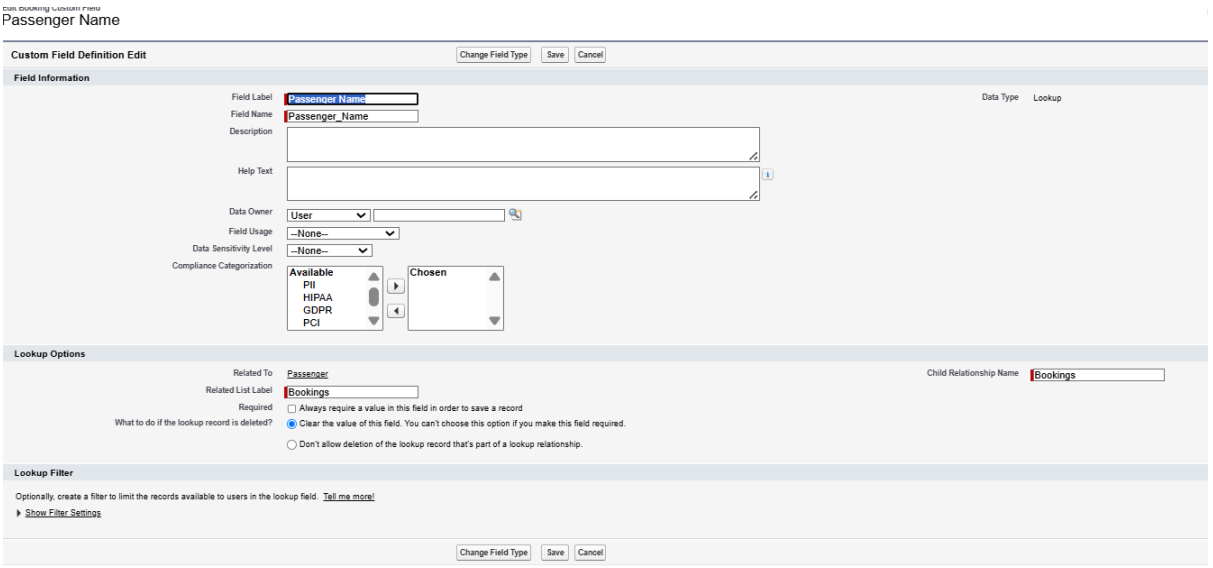
Added essential custom fields: Date, Time, Number, Picklist, Auto Number.

Established lookup relationships:

Booking → Flight

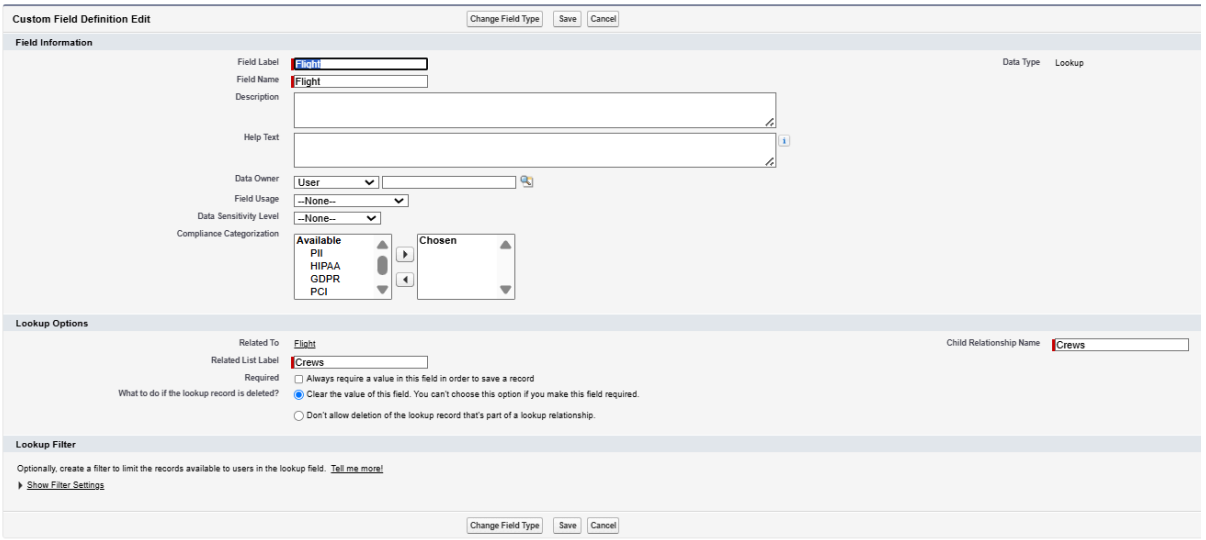


Booking → Passenger



Crew → Flight

Flight

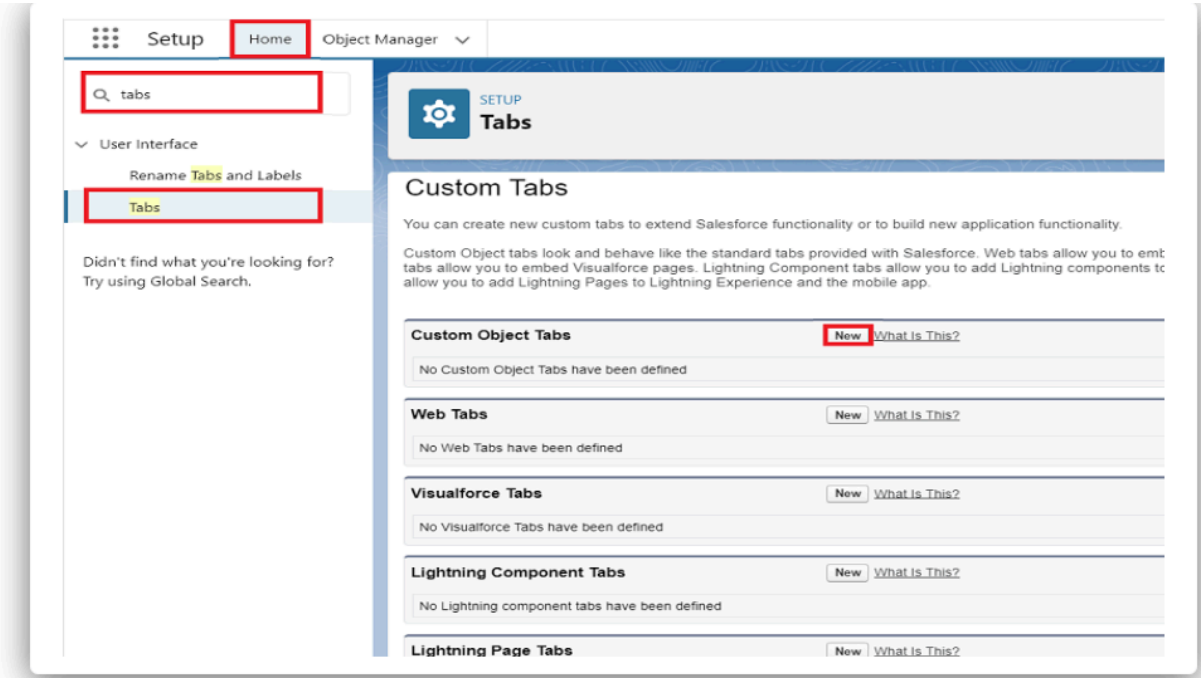


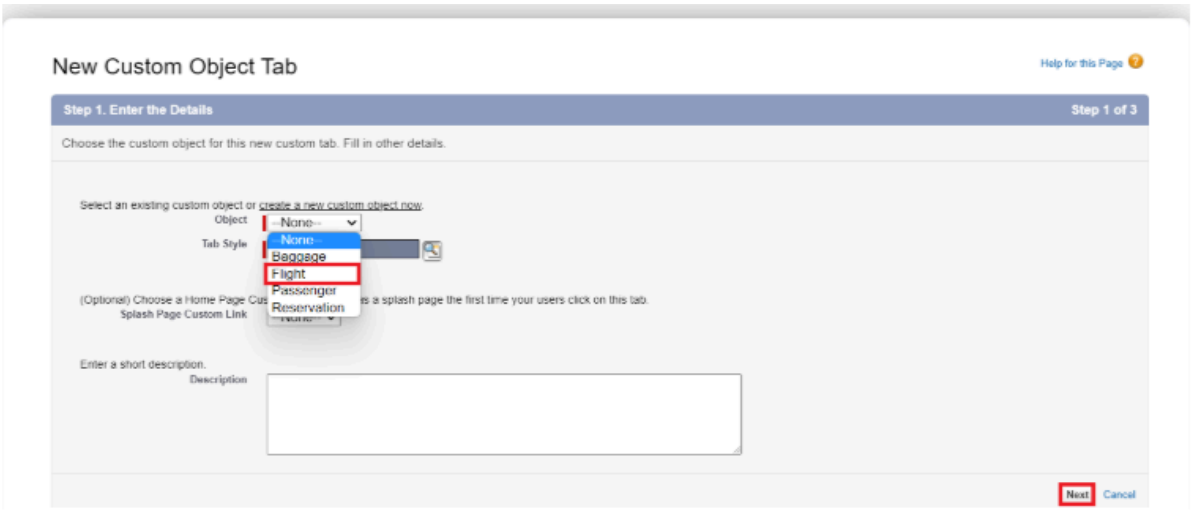
Created field dependencies (e.g., Departs From → Departs To).

6.4 Tab and App Configuration

Developed a Lightning App called Airline Management System.

Added custom tabs for quick access to Flight, Booking, Passenger, and Crew objects.

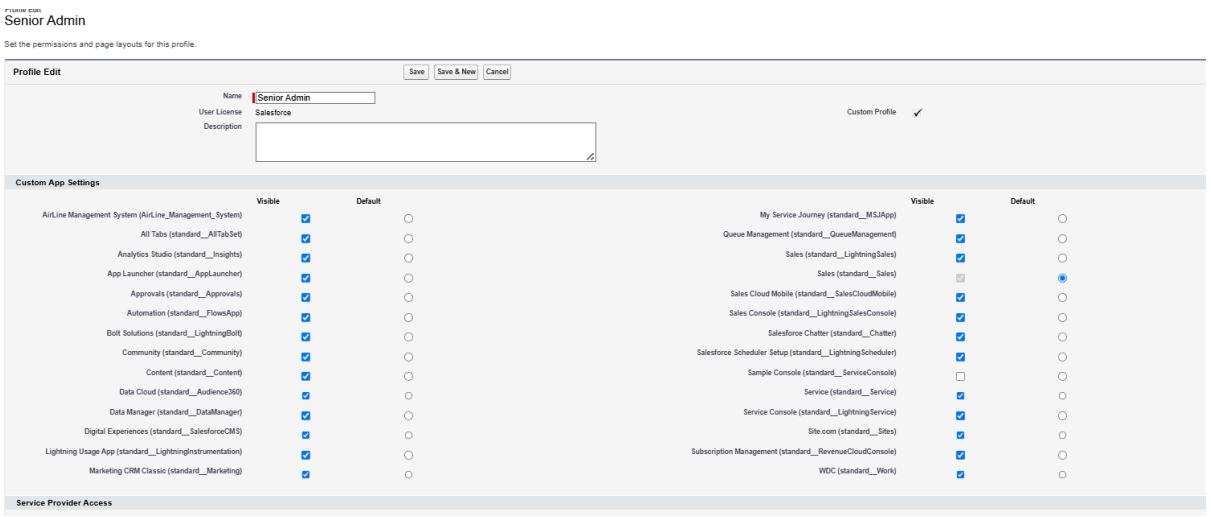

Setup
Home
Object Manager
tabs
User Interface
Rename Tabs and Labels
Tabs
Didn't find what you're looking for?
Try using Global Search.
SETUP
Tabs
Custom Tabs
You can create new custom tabs to extend Salesforce functionality or to build new application functionality.
Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to emb
tabs allow you to embed Visualforce pages. Lightning Component tabs allow you to add Lightning components to
allow you to add Lightning Pages to Lightning Experience and the mobile app.
Custom Object Tabs
New
What Is This?
No Custom Object Tabs have been defined
Web Tabs
New
What Is This?
No Web Tabs have been defined
Visualforce Tabs
New
What Is This?
No Visualforce Tabs have been defined
Lightning Component Tabs
New
What Is This?
No Lightning component tabs have been defined
Lightning Page Tabs
New
What Is This?

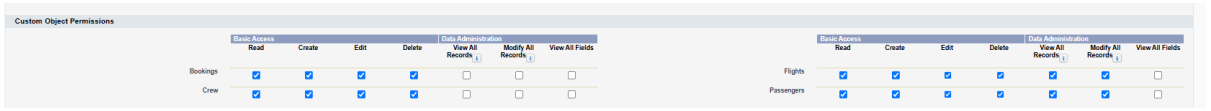

New Custom Object Tab
Help for this Page
Step 1. Enter the Details
Step 1 of 3
Choose the custom object for this new custom tab. Fill in other details.
Select an existing custom object or create a new custom object now.
Object
--None--
Tab Style
--None--
Baggage
Flight
Passenger
Reservation
(Optional) Choose a Home Page Cus
s a splash page the first time your users click on this tab.
Splash Page Custom Link
Enter a short description.
Description
Next
Cancel

6.5 User Interface & Access Control

Designed Profiles:

Senior Admin

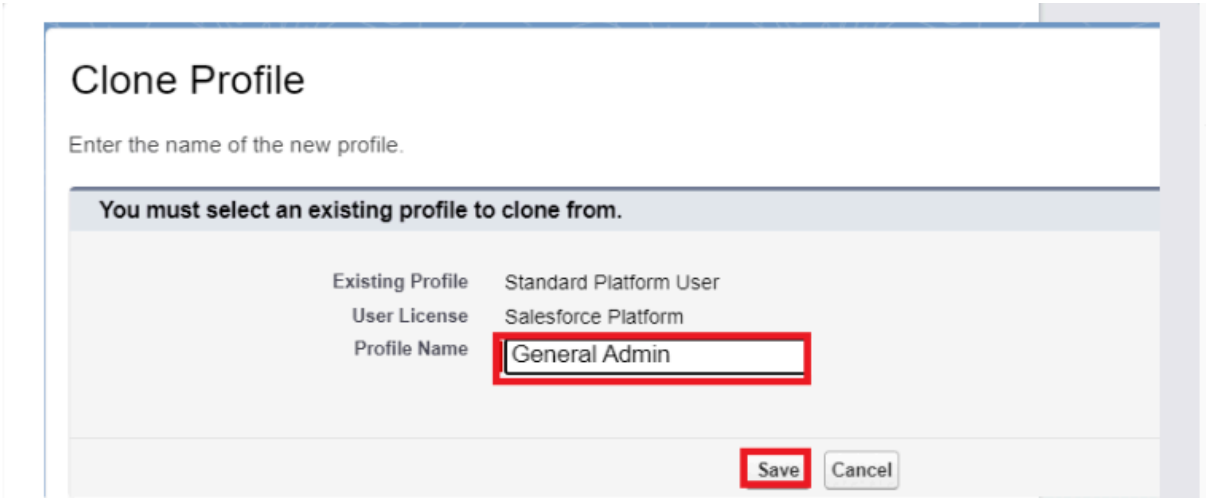




General Admin

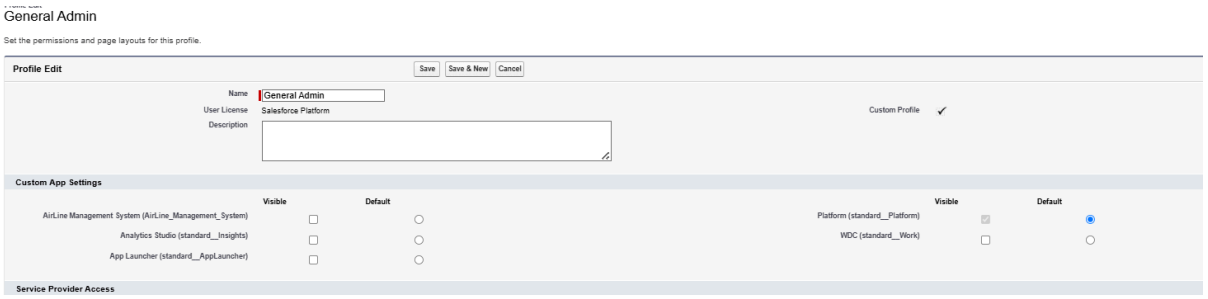
To create a new profile:

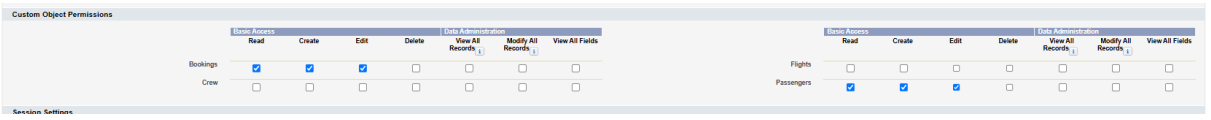
Go to setup --> type profiles in quick find box --> click on profiles --> clone the desired profile (Standard Platform User)-->enter profile name (General Admin) --> Save.



While still on the profile page, then click Edit.

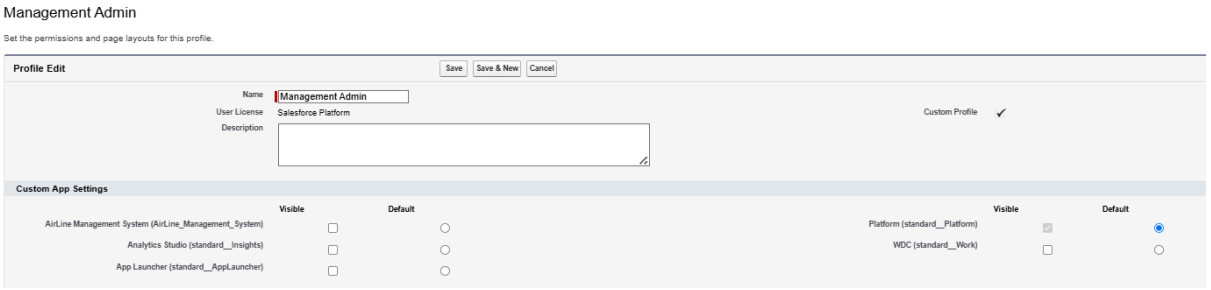
Scroll down to Custom Object Permissions and Give access permissions for read, create and edit on Passenger, and booking objects.

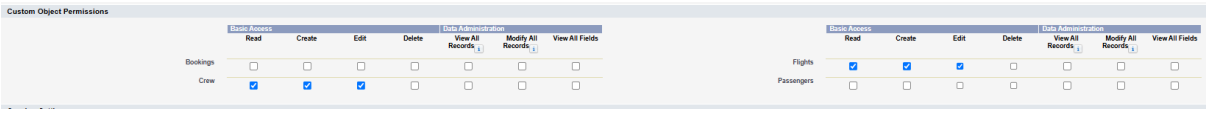




Management Admin

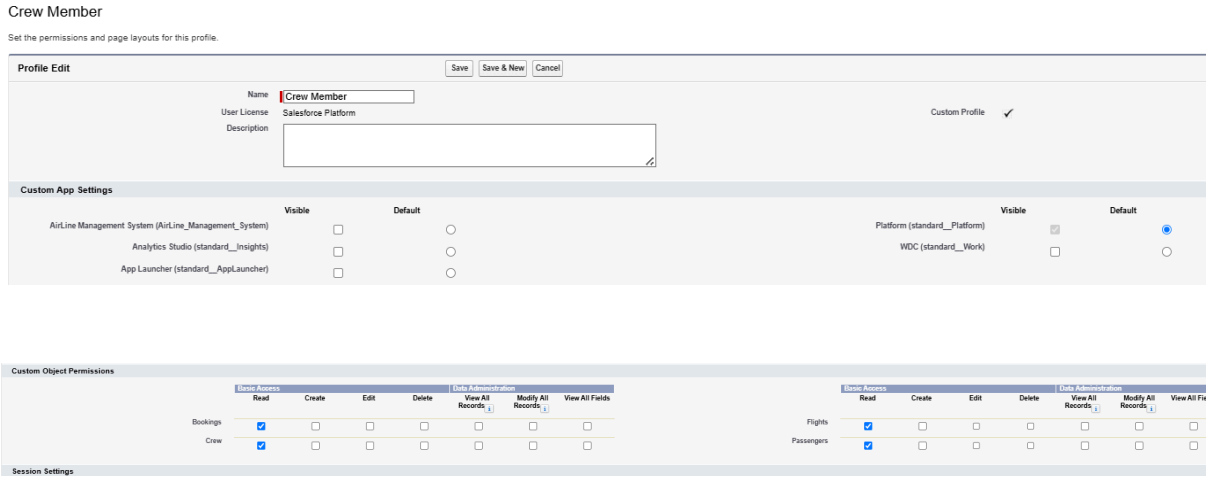
Went to Setup, typed "profiles" in the quick find box, clicked on Profiles, cloned the desired profile (Standard Platform User), entered the profile name (Management Admin), and saved. While still on the profile page, clicked Edit, scrolled down to Custom Object Permissions, gave access permissions for read, edit, and create for Flight and Crew, scrolled down, and clicked on Save.





Crew Member

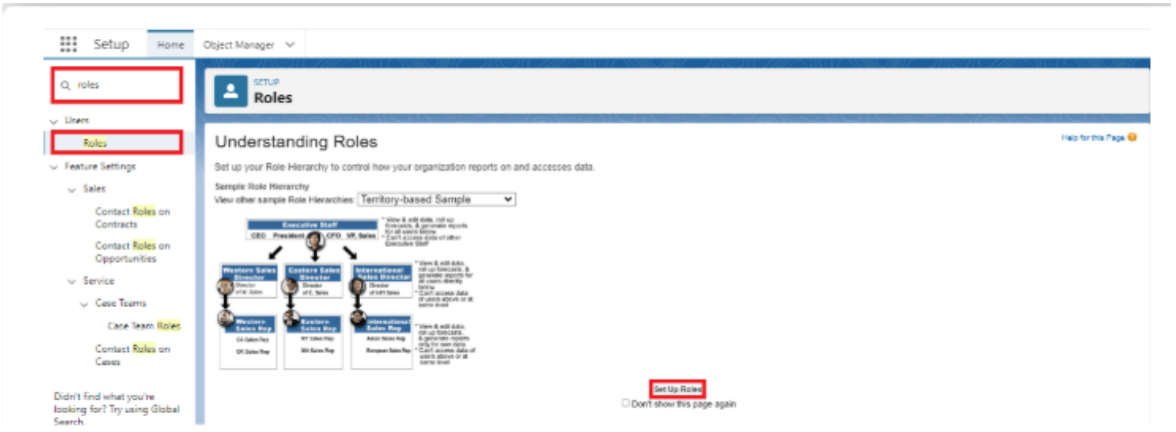
Created Crew Member Profile for Crew Members and gave them access permission to read only as we had done in the previous activity.



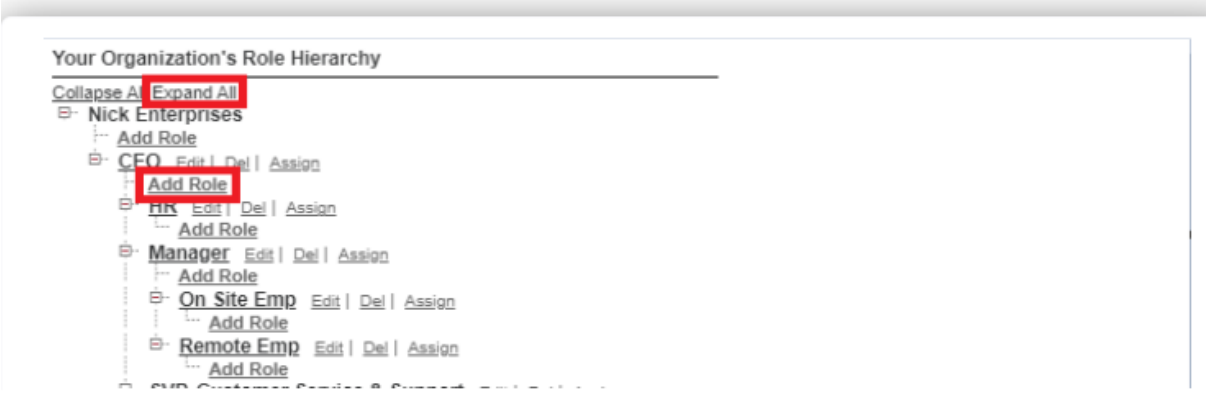
Assigned object permissions based on role requirements.

Created Roles and established hierarchy (Crew Member → Management Admin → Senior Admin → CEO).

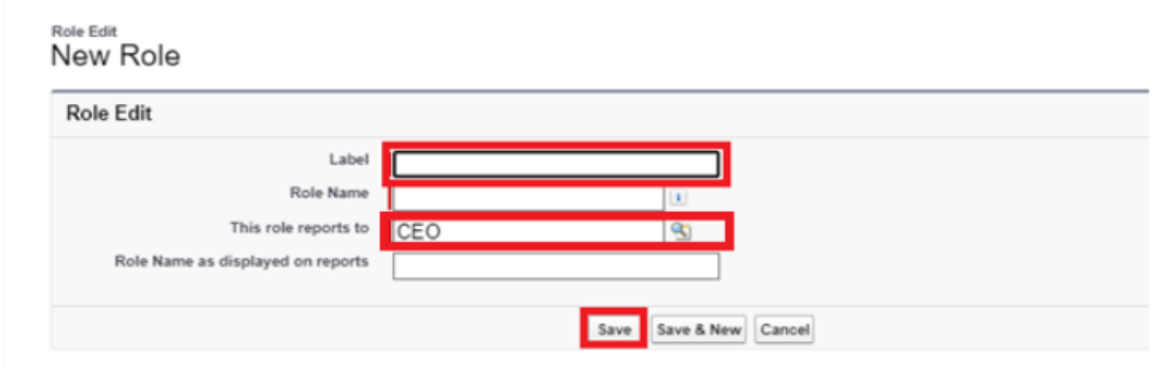
Go to quick find --> Search for Roles --> click on set up roles.



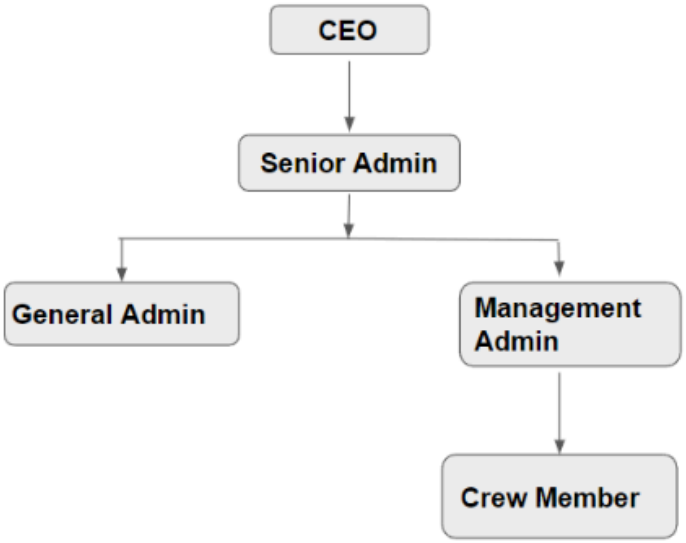
Click on Expand All and click on add role under whom this role works.



Gave the label as “Senior Admin” and the Role name got auto populated. Checked to whom this role (Senior Admin) reported. Then clicked on Save.



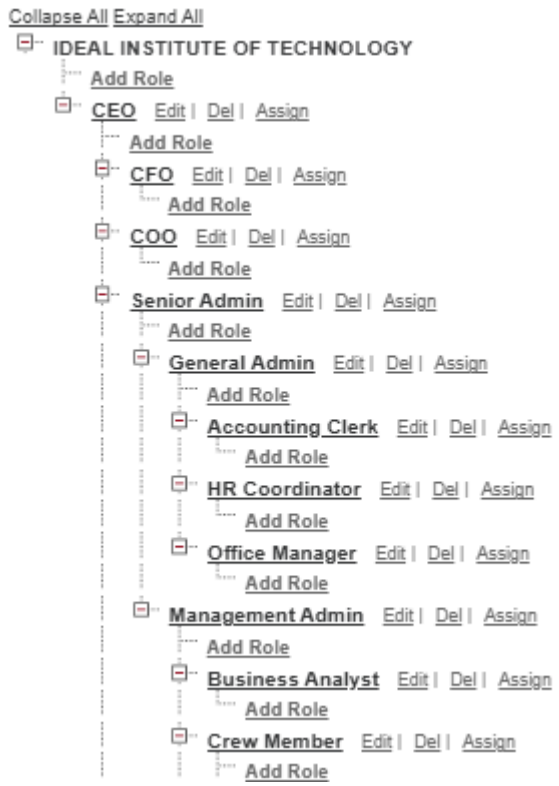
Referred the below diagram to understand which role reports to which role.



Role Hierarchy: The above diagram represents which role reports to which one.

Created three more roles for General Admin, Management Admin and Crew Member.

Note: Crew Members report to Management Admin, Management Admin & General Admin Reports to Senior Admin and only Senior Admin reports to CEO.



6.6 Automation Implementation

Created Apex Trigger to enforce mandatory phone number input for Passenger records.

Built Apex Class and Test Class for validation logic and code coverage.

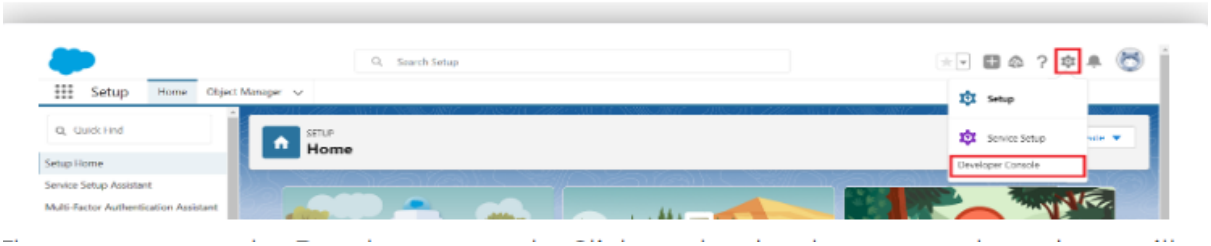
Designed Screen Flow for Booking creation with:

Input screen

Record creation element

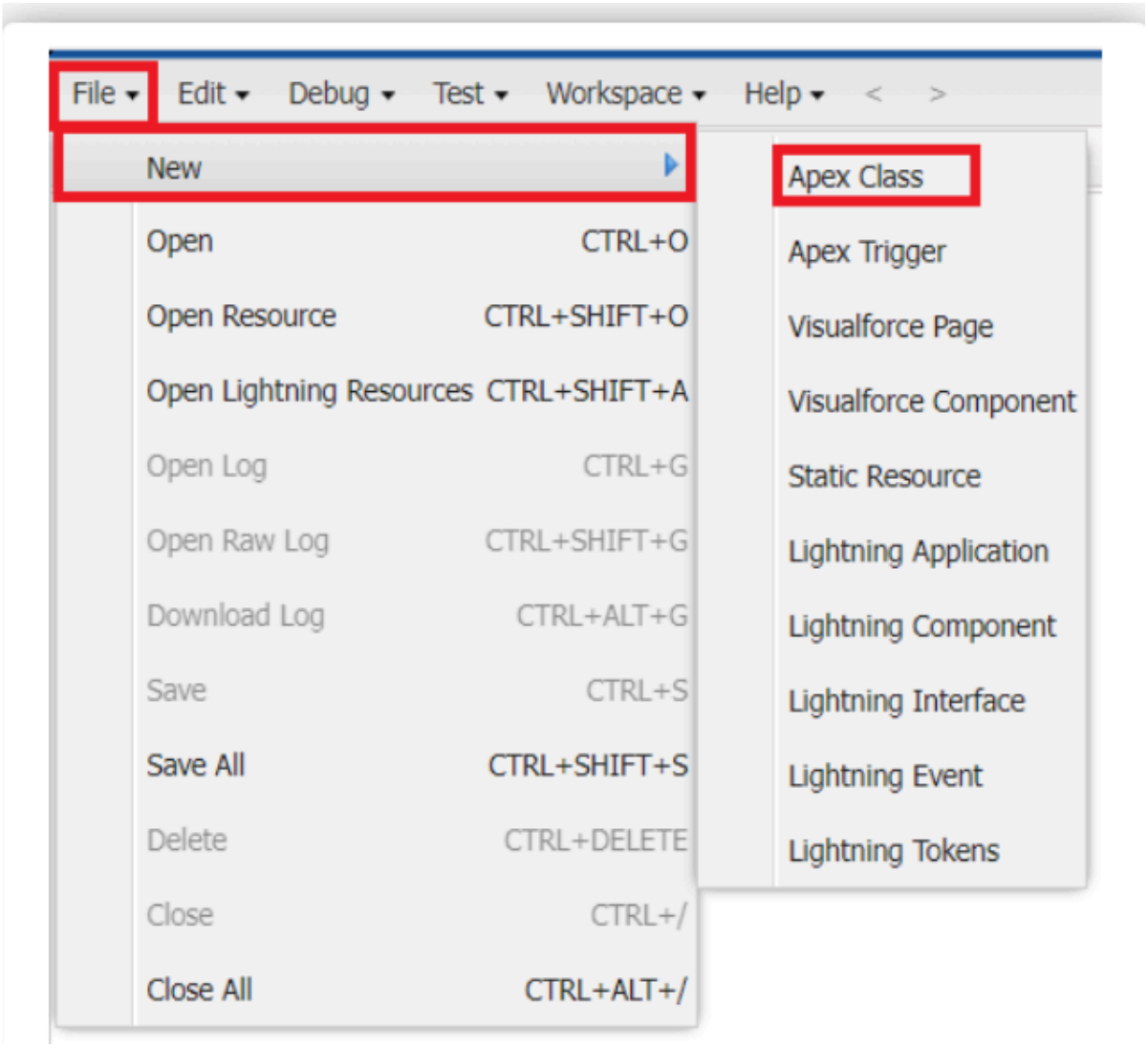
Success confirmation screen

Go to Setup --> Click on the gear icon --> Select Developer Console.

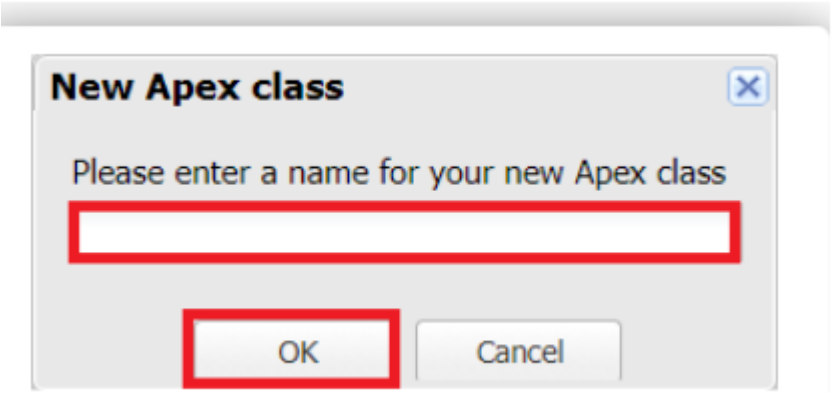


Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.

To create a new Apex Class follow the below steps:
Click on the file --> New --> Apex Class.



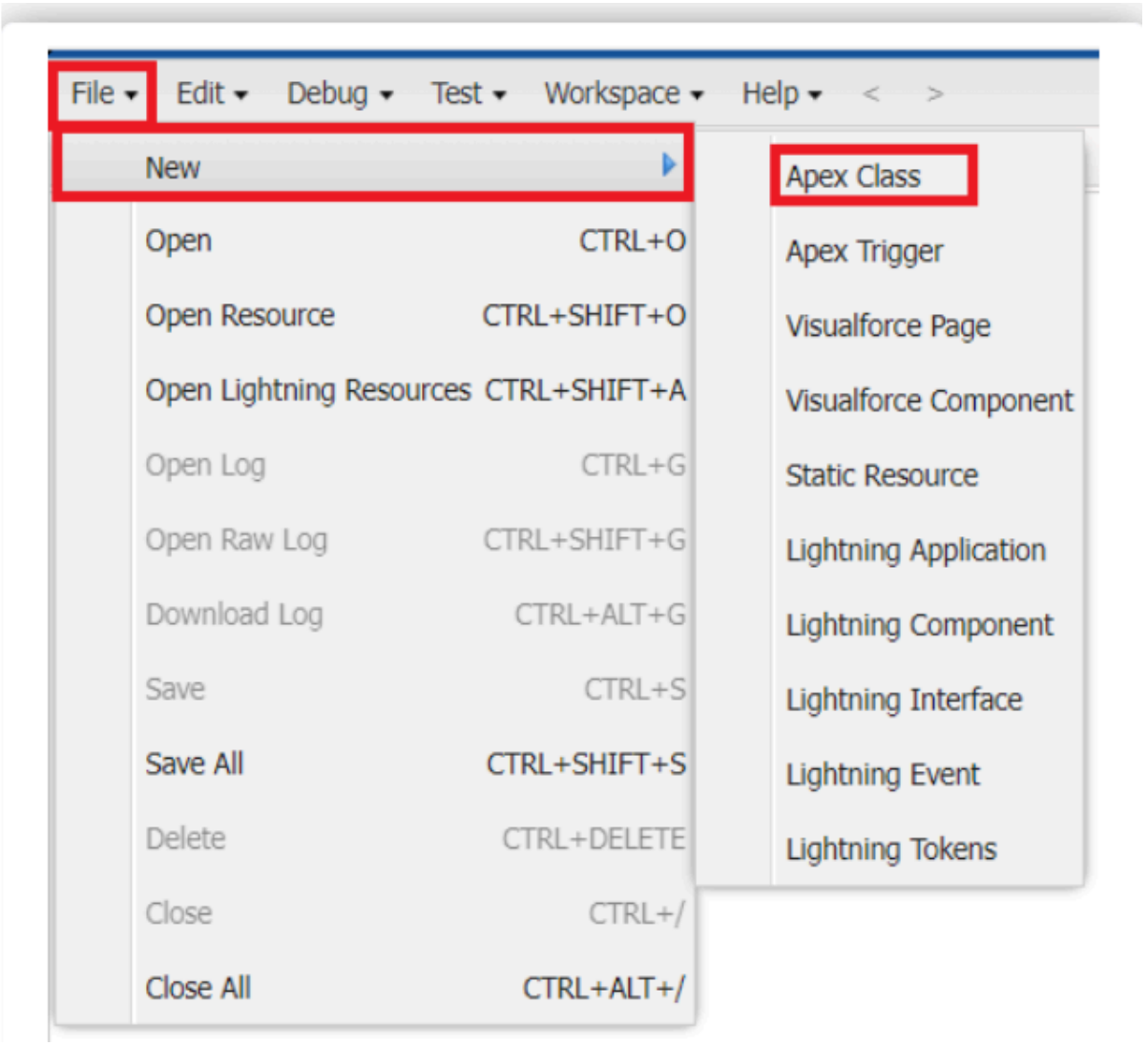
Give the Apex Class name as “PhnValid_PassengerObj”.



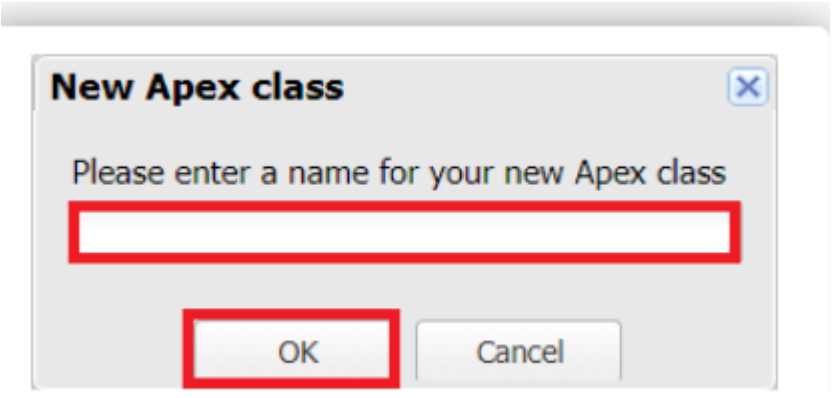
Now write the code logic here

Create an Apex Test Class

To create a new Apex Class follow the below steps:
Click on the file --> New --> Apex Class.

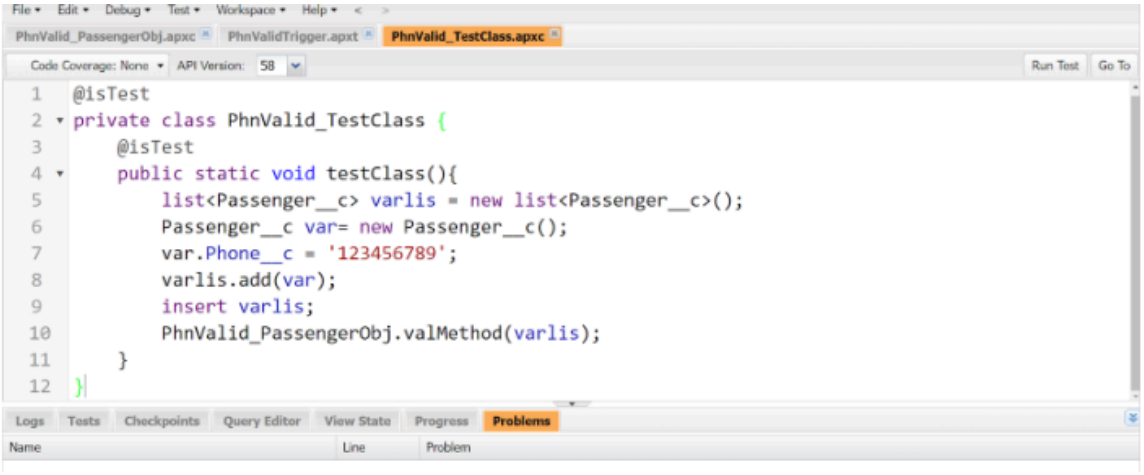


Gave the Apex Class name as “PhnValid_TestClass”.



lick ok

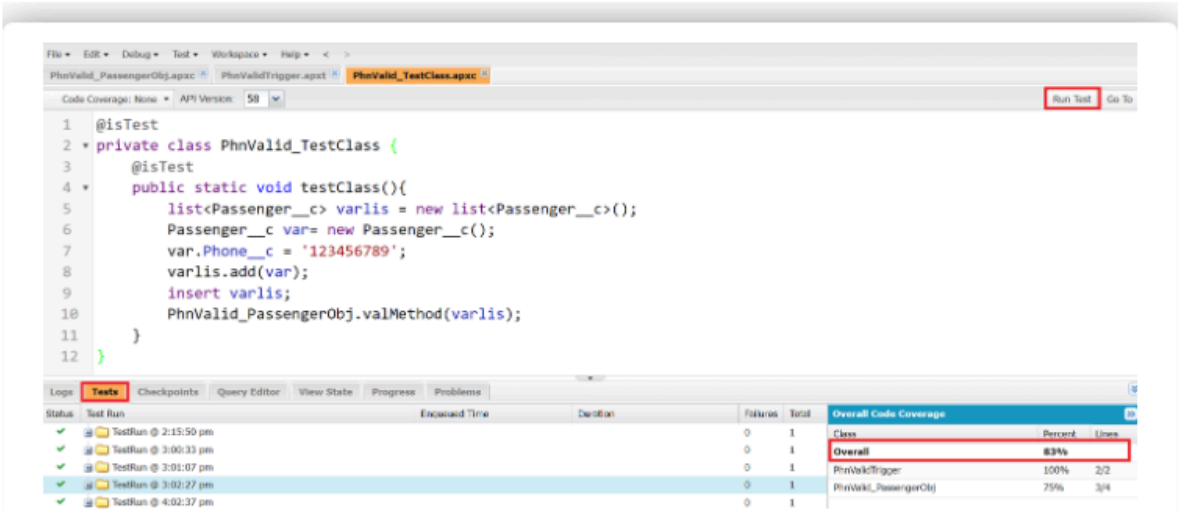
Now wrote the code logic here



```
@isTest
private class PhnValid_TestClass {
    @isTest
    public static void testClass(){
        list<Passenger__c> varlis = new list<Passenger__c>();
        Passenger__c var= new Passenger__c();
        var.Phone__c = '123456789';
        varlis.add(var);
        insert varlis;
        PhnValid_PassengerObj.valMethod(varlis);
    }
}
```

Saved the code.(click on file --> Save).

Clicked on “Run Test” and then clicked on Test under the terminal section and done check that your overall code coverage is more than 75%.



```
@isTest
private class PhnValid_TestClass {
    @isTest
    public static void testClass(){
        list<Passenger__c> varlis = new list<Passenger__c>();
        Passenger__c var= new Passenger__c();
        var.Phone__c = '123456789';
        varlis.add(var);
        insert varlis;
        PhnValid_PassengerObj.valMethod(varlis);
    }
}
```

6.7 Reporting & Dashboards

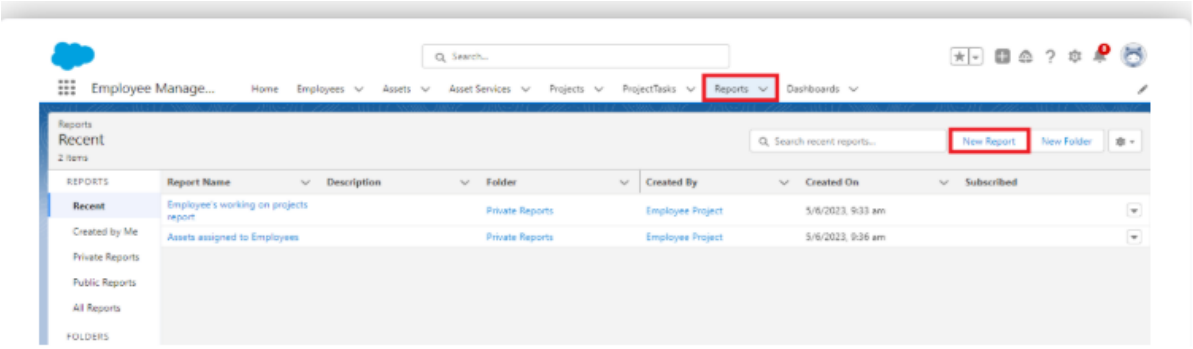
Developed three key Reports:

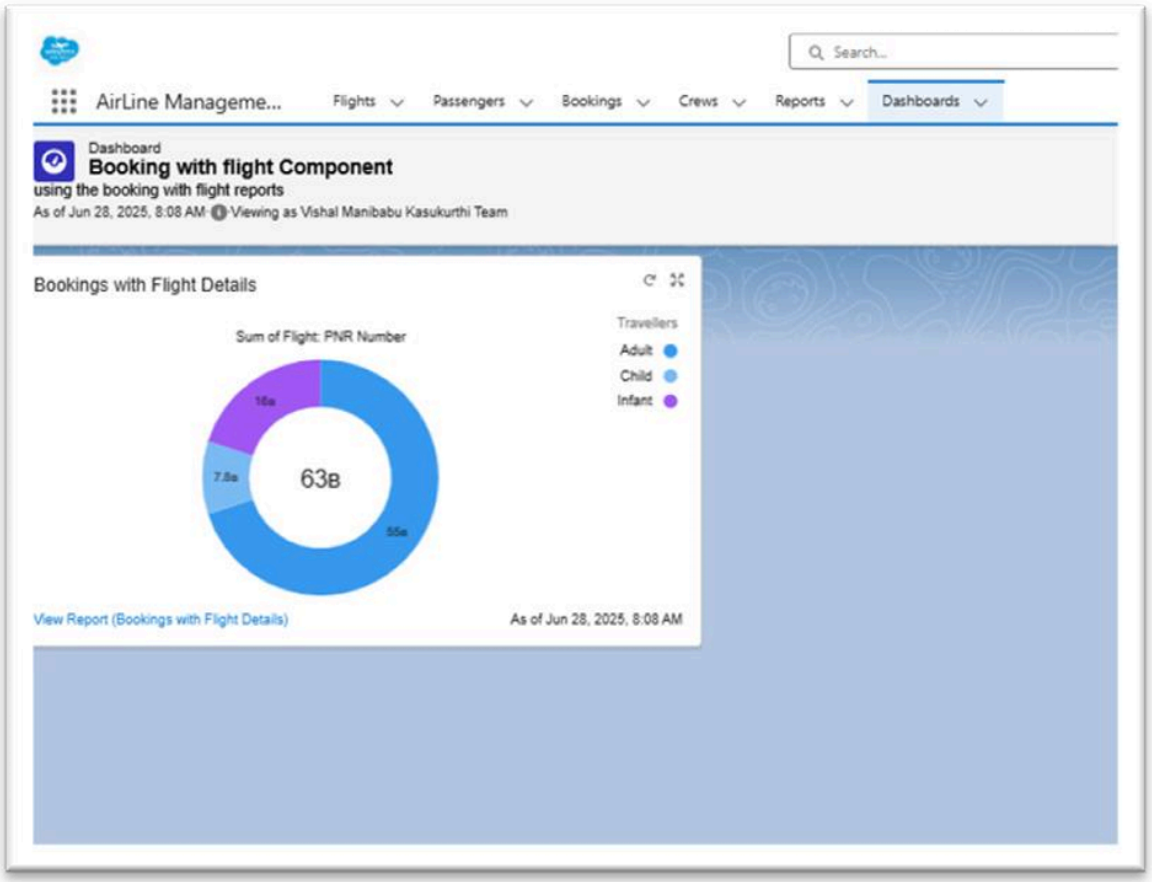
Bookings with Flight Details

Created Report

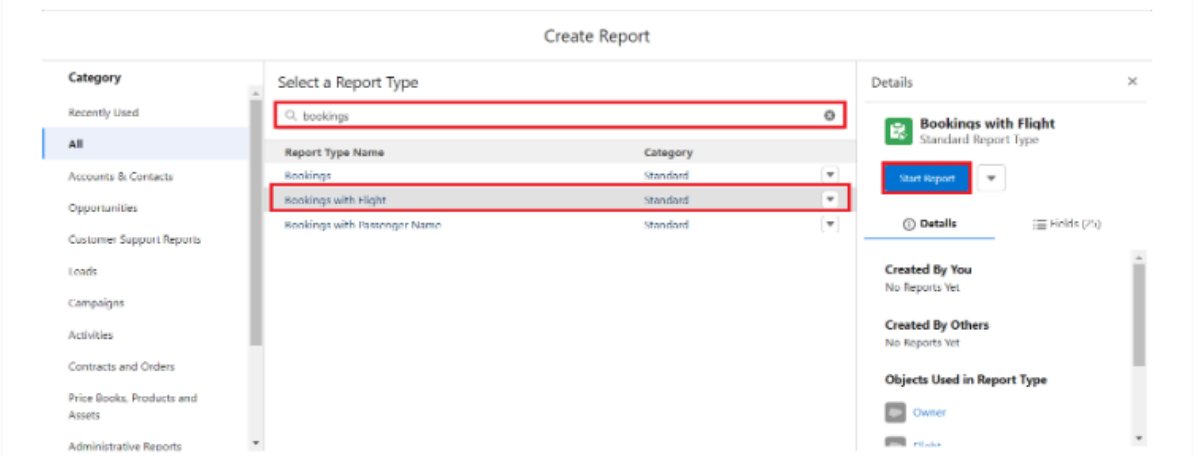
Go to the app --> click on the reports tab

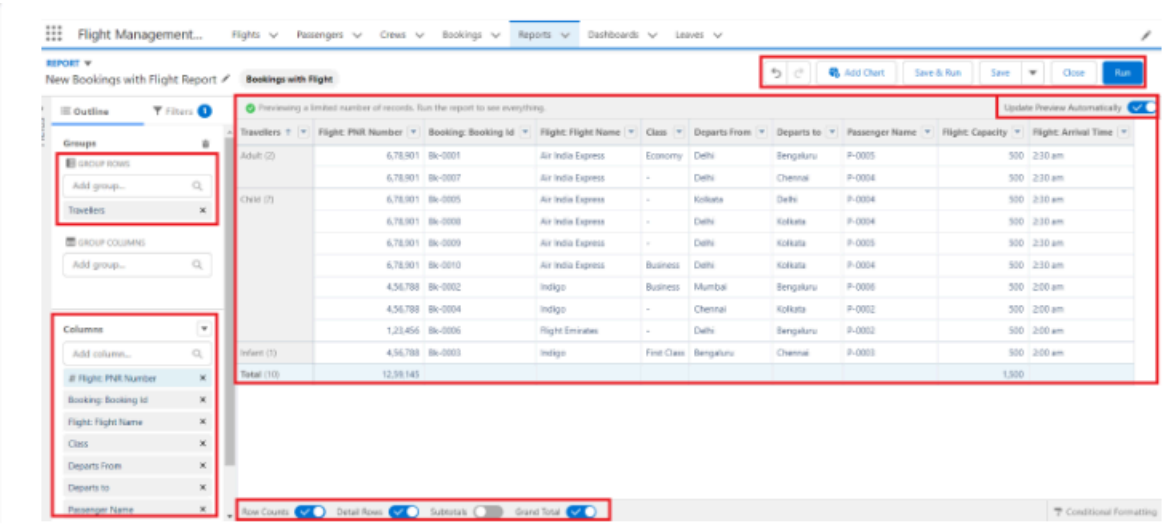
Click New Report.



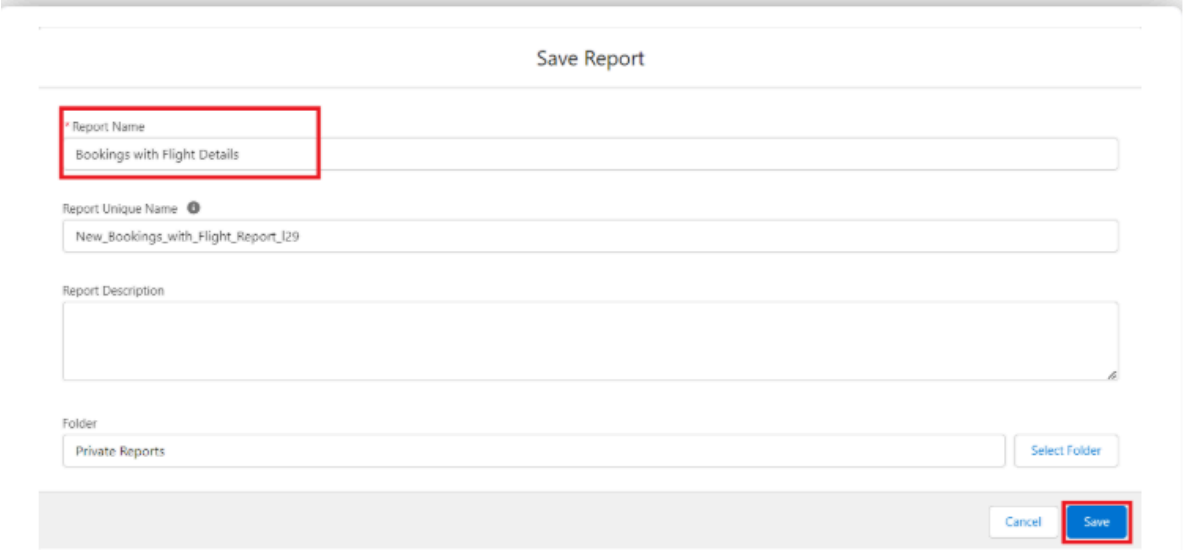


Selected report type from category or from report type panel or from search panel --> clicked on start report.





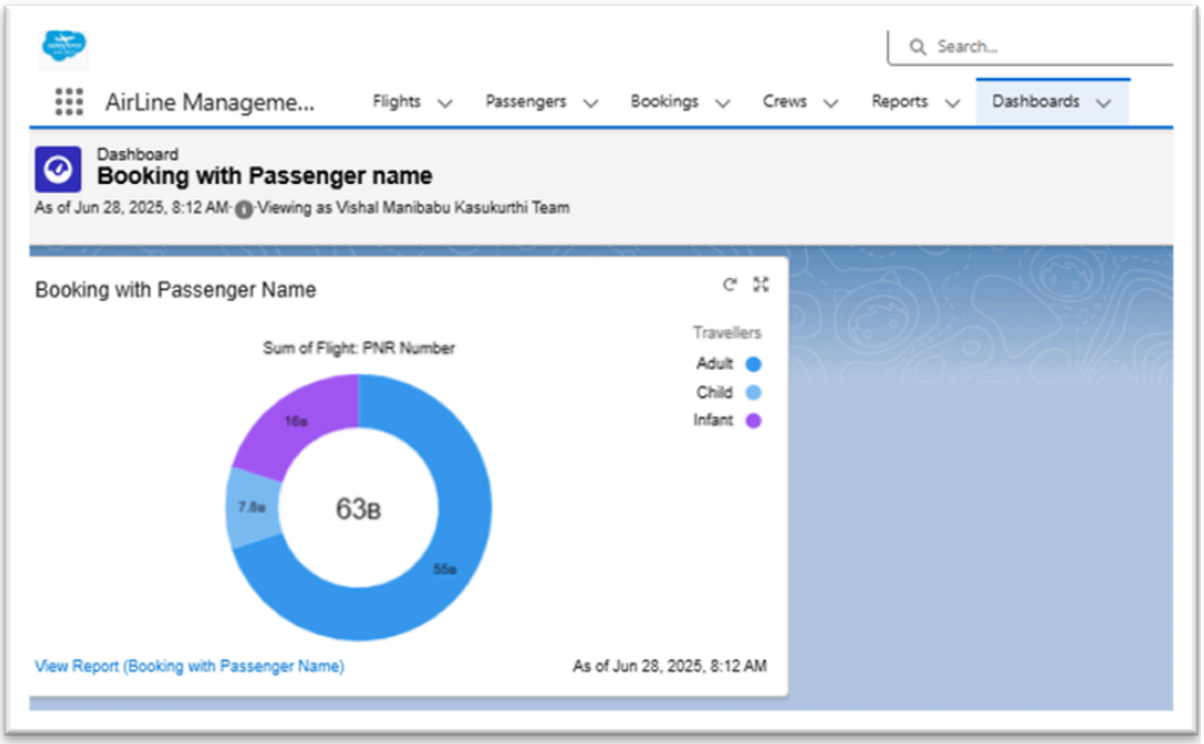
Saved the report as “Bookings with Flight Details”.

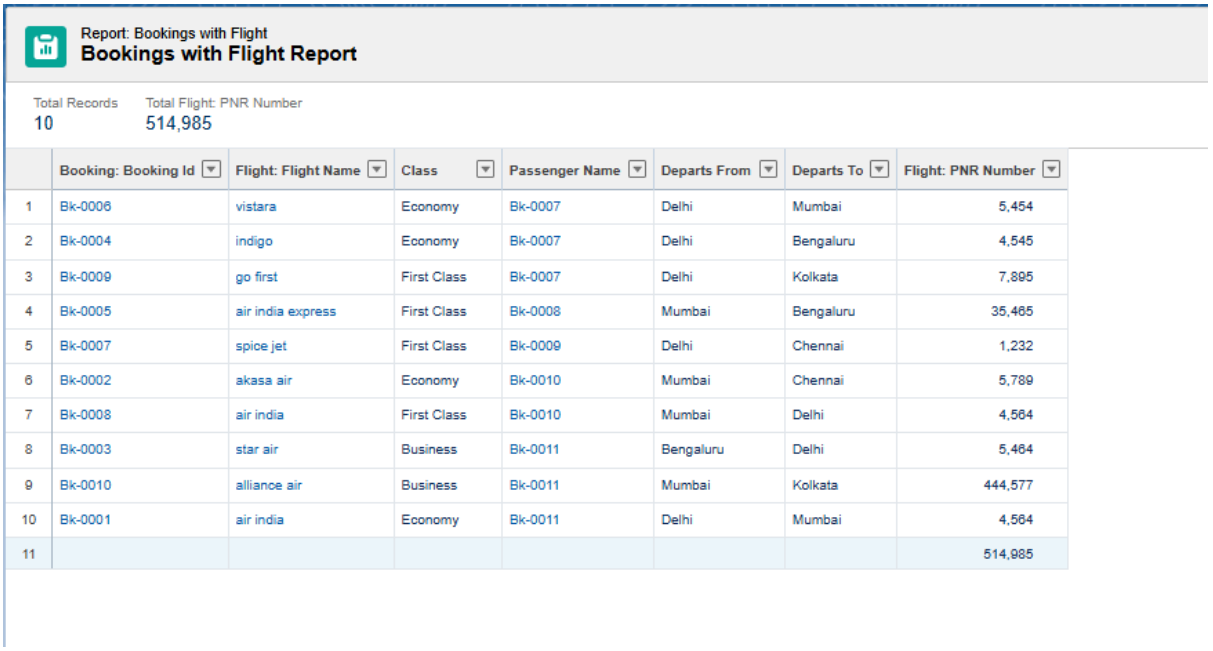


Click on Save.

Bookings with Passenger Names

➤ Created a report to display bookings grouped by Passenger Names for easy analysis and tracking





Report: Bookings with Flight

Bookings with Flight Report

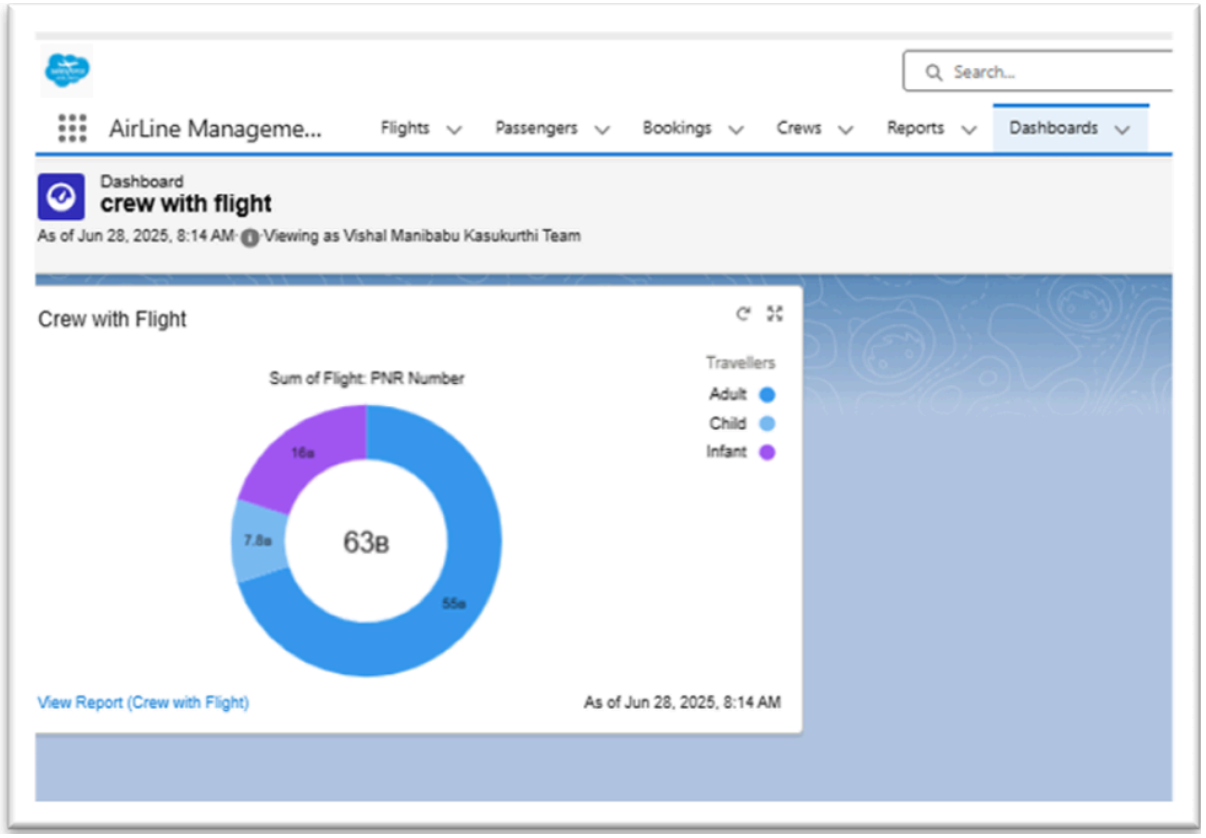
Total Records: 10

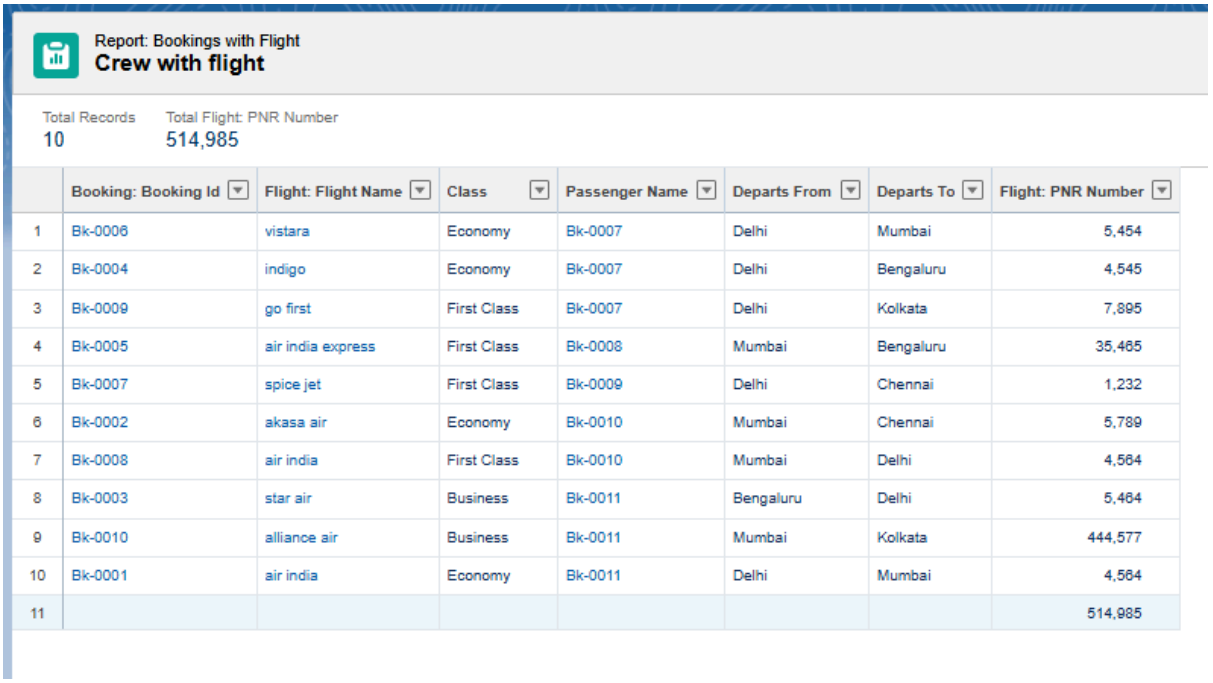
Total Flight: PNR Number: 514,985

	Booking: Booking Id	Flight: Flight Name	Class	Passenger Name	Departs From	Departs To	Flight: PNR Number
1	Bk-0006	vistara	Economy	Bk-0007	Delhi	Mumbai	5,454
2	Bk-0004	indigo	Economy	Bk-0007	Delhi	Bengaluru	4,545
3	Bk-0009	go first	First Class	Bk-0007	Delhi	Kolkata	7,895
4	Bk-0005	air india express	First Class	Bk-0008	Mumbai	Bengaluru	35,465
5	Bk-0007	spice jet	First Class	Bk-0009	Delhi	Chennai	1,232
6	Bk-0002	akasa air	Economy	Bk-0010	Mumbai	Chennai	5,789
7	Bk-0008	air india	First Class	Bk-0010	Mumbai	Delhi	4,564
8	Bk-0003	star air	Business	Bk-0011	Bengaluru	Delhi	5,464
9	Bk-0010	alliance air	Business	Bk-0011	Mumbai	Kolkata	444,577
10	Bk-0001	air india	Economy	Bk-0011	Delhi	Mumbai	4,564
11							514,985

Crew with Assigned Flights

Created a report with report type: “Crew with Flight”





Report: Bookings with Flight
Crew with flight

Total Records: 10
Total Flight: PNR Number: 514,985

	Booking: Booking Id	Flight: Flight Name	Class	Passenger Name	Departs From	Departs To	Flight: PNR Number
1	Bk-0006	vistara	Economy	Bk-0007	Delhi	Mumbai	5,454
2	Bk-0004	indigo	Economy	Bk-0007	Delhi	Bengaluru	4,545
3	Bk-0009	go first	First Class	Bk-0007	Delhi	Kolkata	7,895
4	Bk-0005	air india express	First Class	Bk-0008	Mumbai	Bengaluru	35,465
5	Bk-0007	spice jet	First Class	Bk-0009	Delhi	Chennai	1,232
6	Bk-0002	akasa air	Economy	Bk-0010	Mumbai	Chennai	5,789
7	Bk-0008	air india	First Class	Bk-0010	Mumbai	Delhi	4,564
8	Bk-0003	star air	Business	Bk-0011	Bengaluru	Delhi	5,464
9	Bk-0010	alliance air	Business	Bk-0011	Mumbai	Kolkata	444,577
10	Bk-0001	air india	Economy	Bk-0011	Delhi	Mumbai	4,564
11							514,985

Created corresponding Dashboards using donut and pie charts for visual insights.

6.8 Schema Visualization

Used Schema Builder to visually represent object relationships and confirm database design consistency.

7. FUNCTIONAL AND PERFORMANCE TESTING

7.1 Performance Testing

Data entry tested across 10+ records for each object

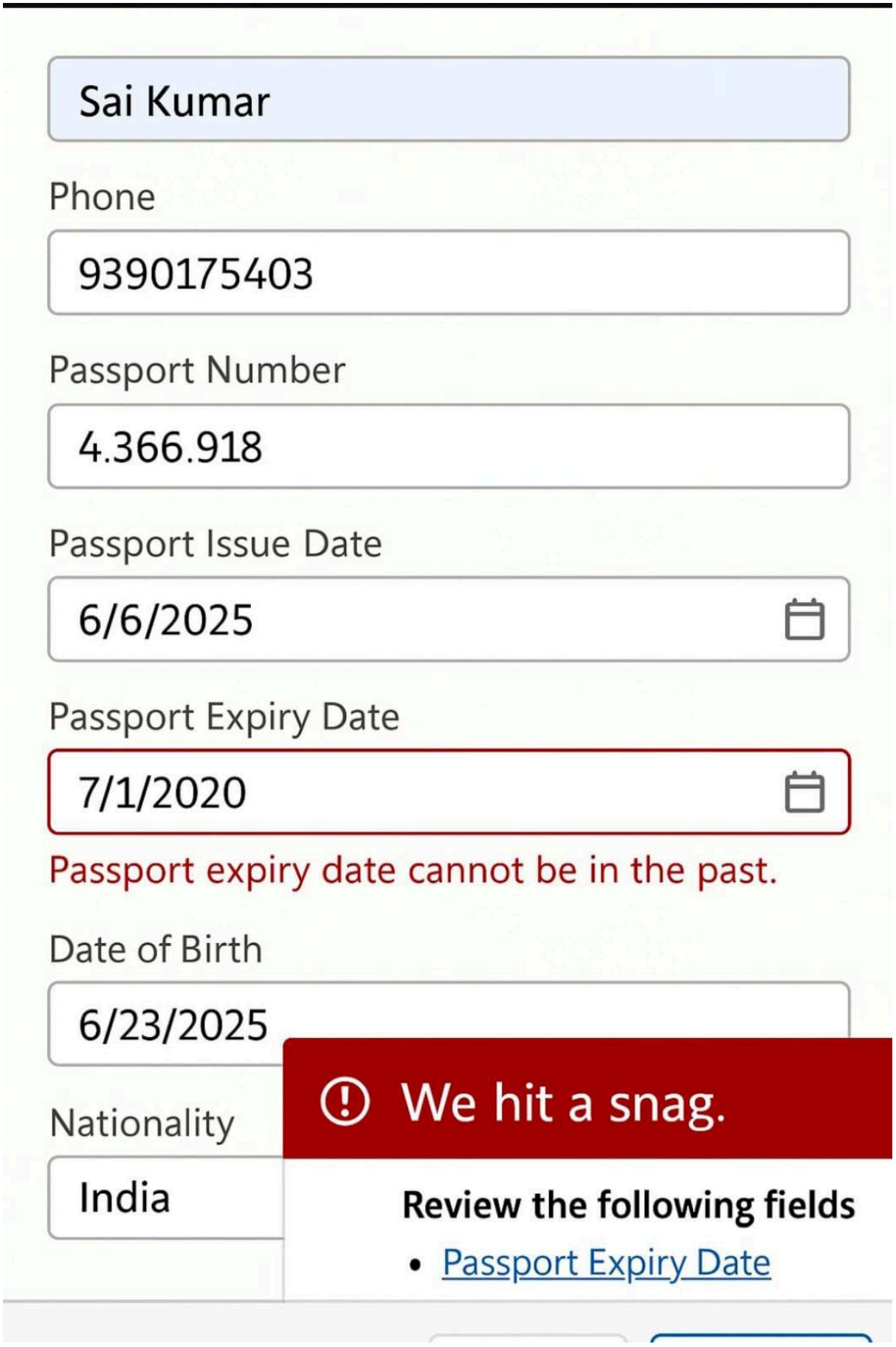
Apex trigger validated for phone field

Flows tested for booking object creation

8. RESULTS

8.1 Output

Screenshots Included in the appendix: Object creation, record insertion, reports, dashboards, flow screens.
Trigger Error on Passport Expiry date


Sai Kumar
Phone
9390175403
Passport Number
4.366.918
Passport Issue Date
6/6/2025
Passport Expiry Date
7/1/2020
Passport expiry date cannot be in the past.
Date of Birth
6/23/2025
Nationality
India
We hit a snag.
Review the following fields
Passport Expiry Date

Trigger Validation Error When Passport Expiry Date Is in the Past

To ensure data accuracy, I implemented a before insert and before update Apex trigger that prevents a **Passenger** record from being saved if the **Passport Expiry Date** is in the past. This validation was handled using a custom Apex class and tested with both valid and invalid expiry dates.

Class Name: PassportExpiryValidator.cls

```
public class PassportExpiryValidator {

    public static void validate(List<Passenger__c> passengers) {

    for (Passenger__c p : passengers) {

    if (p.Passport_Expiry_Date__c != null &&
p.Passport_Expiry_Date__c < Date.today()) {

        p.Passport_Expiry_Date__c.addError('Passport expiry date
cannot be in the past.');

    }

    }

    }

}
```

✔️ This class checks each record's Passport Expiry Date and adds an error if the date is earlier than today.

Step 2: Created Apex Trigger

```
trigger PassportExpiryTrigger on Passenger__c (before insert, before
update) {
```

```
    if (Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
        PassportExpiryValidator.validate(Trigger.new);
}
}
```

✔️ This trigger runs **before insert and update**, and calls the validation method from the class.

Here is a **fully adapted version** of your original Lease Management System documentation — now modified to match your **Airline Management System (AMS)** project. This includes changes to advantages, future scope, conclusion, and code examples, all tailored for an airline domain..

9. ADVANTAGES & DISADVANTAGES

Advantages:

Automated operations

Real-time data access

User-role customization

Visual analytics with dashboards

Disadvantages:

Initial setup complexity

Dependent on internet access

10. CONCLUSION

The Salesforce-powered Airlines Management System brings structure, automation, and data integrity to airline operations, enabling better decision-making and improved service.

11. FUTURE SCOPE

Integration with third-party APIs (e.g., payment gateways)

Mobile app development

AI-powered analytics

12. APPENDIX

Apex Class: PhnValid_PassengerObj.cls

```
public class PhnValid_PassengerObj
{
  public static void valMethod(List newPass)
 {
  for(Passenger__c p : newPass)
  {
   if(p.Phone__c == null)
   {
```

```
    p.Phone__c.addError('please Enter phone Number');
   }
 }
}
```

Apex Trigger: PhnValidTrigger.trigger

```
trigger PhnValidTrigger on Passenger__c (before insert) {
        if(trigger.isBefore && trigger.isInsert){
            PhnValid_PassengerObj.valMethod(trigger.new);
         }
}
```

Apex Test Class: PhnValid_TestClass.cls

```
        @isTest
        public class PhnValid_TestClass {
            @isTest
            public static void testClass(){
                list <Passenger__c> varlis = new
        list<Passenger__c>();
                Passenger__c var = new Passenger__c();
                var.Phone__c = null;
                varlis.add(var);
                insert varlis;
                PhnValid_PassengerObj.valMethod(varlis);
            }
        }
```

DEMO VIDEO:LINK:https://drive.google.com/file/d/1ybPk0ax_3nZp6-KlXt7S6-MjXEkLAZ8l/view?usp=sharing

GitHub:https://github.com/Sai-Kumar89654/AIRLINE-MANAGEMENT-SYSTEM